AFRL-RH-WP-TR-2021-0113

SYSTEM FOR CROSS-LANGUAGE INFORMATION PROCESSING, TRANSLATION AND SUMMARIZATION (SCRIPTS)

Kathleen McKeown / Julia Hirschberg
Smaranda Muresan / Ramy Eskander
Faisal Ladhak / Susan McGregor
Victor Soto Martinez / Elsbeth Turcan
David Wan
Columbia University
Dept of Computer Science
500 W 120th
New York, NY 10027

Doublas Oard / Marine Carpuat
Joseph Barrow /Petra Galuščáková
Suraj Nair / Xin Niu / Peter Rankel Aquia
Richburg / Yow-Ting Shiue
Han-Chin Shing / Weijia Xu / Elena Zotkina
University of Maryland
College of Info Studies
4130 Campus Drive
College Park, MD 20742

Peter Bell / Kenneth Heafield
Ojdrej Klejch / Sukanta Sen
Svetlana Tshistiakova
Electra Wallington
University of Edinburgh
Informatics Forum
10 Crichton Street
Edinburgh UK EH 9AB

Dragomir Radev / Neha Verma
Rui Zhang
Yale University
Dept of Computer Science
51 Prospect St.
New Haven, CT 06511

Mark J.F. Gales / Kate Knill
Xinxin Wu
University of Cambridge
Information Engineering Division
Trumpington, Cambridge UK
CB2 1PZ

December 2021

Final Report

Distribution A. Approved for public release; distribution unlimited.

AIR FORCE RESEARCH LABORATORY
711TH HUMAN PERFORMANCE WING
AIRMAN SYSTEMS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE OH 45433
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

NOTICE AND SIGNATURE PAGE

AFRL-RH-WP-TR-2021-0113 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

ANDERSON.TIMOTHY.RAY.1230210728 Digitally signed by ANDERSON.TIMOTHY.RAY.1230210728 Date: 2023.01.05 13:02:22 -05'00'

TIMOTHY R. ANDERSON, DR-IV, Ph.D.
Work Unit Manager
Mission Analytics Branch
Airman Systems Directorate
711th Human Performance Wing
Air Force Research Laboratory

MURDOCK.WILLIAM.P.1048742161 Digitally signed by MURDOCK.WILLIAM.P.1048742161 Date: 2023.01.05 14:05:58 -05'00'

WILLIAM P. MURDOCK, DR-IV, Ph.D.
Chief, Mission Analytics Branch
Airman Systems Directorate
711th Human Performance Wing
Air Force Research Laboratory

CARTER.LOUISE.ANN.1230249128 Digitally signed by CARTER.LOUISE.ANN.1230249128 Date: 2023.01.05 14:48:22 -05'00'

LOUISE A. CARTER, DR-IV, Ph.D.
Chief, Warfighter Interactions and Readiness Division
Airman Systems Directorate
711th Human Performance Wing
Air Force Research Laboratory

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE	**2. REPORT TYPE**	**3. DATES COVERED**	
		START DATE	**END DATE**
12/22/2021	Final	09/25/2017	12/22/2021

4. TITLE AND SUBTITLE

System for Cross-Language Information Processing, Translation and Summarization (SCRIPTS)

5a. CONTRACT NUMBER FA8650-17-C-9117	**5b. GRANT NUMBER**	**5c. PROGRAM ELEMENT NUMBER**
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER** H0V1

6. AUTHOR(S) a) Kathleen McKeown/Julia Hirschberg/Smaranda Muresan/Ramy Eskander/Faisal Ladhak/Susan McGregor/Victor Soto Martinez/Elsbeth Turcan/David Wan b) Doublas Oard / Marine Carpuat/Joseph Barrow/Petra Galuščáková/Suraj Nair/Xin Niu/Peter Rankel/Aquia Richburg / Yow-Ting Shiue Han-Chin Shing/Weijia Xu/Elena Zotkina c) Peter Bell/Kenneth Heafield/Ojdrej Klejch/Sukanta Sen/Svetlana Tshistiakova/Electra Wallington d) Dragomir Radev/Neha Verma/Rui Zhang e) Mark J.F. Gales/Kate Knill/Xinxin Wu

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) a) Columbia University, Dept. of Computer Science, 500 W 120th St, New York NY 10027 b) Univ. of Maryland, College of Info. Studies, 4130 Campus Drive, College Park MD 20742 c) Univ. of Edinburgh, Informatics Forum, 10 Crichton Street, Edinburgh, UK EH8 9AB, UK d) Yale University, Dept of Computer Science, 51 Prospect St, New Haven, CT 06511 e) Univ. of Cambridge, Information Eng. Div., Trumpington, Cambridge, UK CB2 1PZ	**8. PERFORMING ORGANIZATION REPORT NUMBER**

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory 711th Human Performance Wing Airman Systems Directorate Warfighter Interactions and Readiness Division Wright-Patterson Air Force Base, OH 45433	**10. SPONSOR /MONITOR'S ACRONYM(S)**	**11. SPONSOR/MONITOR'SREPORT NUMBER(S)** AFRL-RH-WP-TR-2021-0113

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution A. Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

AFRL-2022-5072; Cleared 20 Oct 2022

14. ABSTRACT

This report describes the technical approaches and results for System for Cross-language Information Processing, Translation and Summarization (SCRIPTS) funded under the IARPA MATERIAL program. SCRIPTS consists of components for Automatic Speech Recognition (ASR) and Machine Translation (MT) in order to preprocess the text and speech corpora provided as part of the program. It also includes a text processing component that performs morphological analysis. In user-facing mode, given a query, SCRIPTS' Cross-Language Information Retrieval (CLIR) returns relevant documents, while Summarization generates textual summaries of each document to help an analyst confirm which documents returned by CLIR are actually relevant. Over the course of program, the team implemented models for nine different languages: Somali, Swahili, Tagalog, Bulgarian, Lithuanian, Pashto, Farsi, Kazakh, and Georgian.

15. SUBJECT TERMS

Cross-lingual information retrieval (CLIR), machine translation (MT), automatic speech recognition (ASR), summarization, morphology, statistical modeling, machine learning

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**		
U	U	U	SAR	191

19a. NAME OF RESPONSIBLE PERSON Timothy R. Anderson, Ph.D.	**19b. PHONE NUMBER** *(Include area code)*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

This report describes our technical approaches and results for System for CRoss-language Information Processing, Translation and Summarization (SCRIPTS) funded under the Intelligence Advanced Research Projects Activity (IARPA) Machine Translation (MT)for English Retrieval of Information in Any Language (MATERIAL) program. SCRIPTS consists of com-ponents for Automatic Speech Recognition (ASR) and MT in order to pre-process the text and speech corpora provided as part of the program. It also includes a text processing component that performs morphological analysis. In user-facing mode, given a query, SCRIPTS' Cross-Language Information Retrieval (CLIR) returns relevant documents, while Summarization generates textual summaries of each document to help an analyst confirm which documents returned by CLIR are actually relevant. Over the course of the program, we implemented models for nine different languages: Somali, Swahili and Tagalog in the Base Period (BP), Bulgarian, Lithuanian and Pashto in Option Period 1 (OP1), and Farsi, Kazakh and Georgian in OP2.

To address the low resource scenario, our novel approach in SCRIPTS features implicit and explicit integration within and across components. We use a fail-soft approach where each component implements multiple, complementary approaches to the task; these components are then integrated using system combination. For ASR, we developed two approaches, one at Cambridge University (CUED) and one at the University of Edinburgh (EDIN); here, system combination yielded improvements in scores over individual systems. For MT, we developed two neural approaches, one at EDIN and another at the University of Maryland (UMD). UMD also implemented a statistical machine translation (SMT) approach. During evaluations, we ran all three MT systems on the evaluation corpora and all three sets of results were saved and used by both CLIR and Summarization. CLIR used a large variety of technical approaches and system combination to merge the resulting rankings. While CLIR focused on finding relevant documents given the query, the task for Summarization was to find relevant sentences within the documents returned by CLIR. Like CLIR, Summarization also used multiple approaches for the problem, exploring both unsupervised and supervised methods.

Other key features of our approach include tight interaction between different components. For example, CLIR relies on an interaction with ASR and MT in order to handle search over speech in low resource languages (LRL). Summarization also relies on results from CLIR as well as interaction with MT in order to ensure that at least one of its summary sentences contains the query word(s).

The report is structured as follows. First, we provide a summary of major differences between components across the BP, OP1 and OP2. Then we provide detailed information on the technical approaches taken by each component followed by a section providing results, again segmented by system component. This is followed by a section providing transitionable software. Finally, we close with each team's observations about their work within the MATERIAL program, as well as recommendations for future work based on their research and results.

2.0 INTRODUCTION

The SCRIPTS development teams worked both in tandem and in parallel to develop, test, share and refine their respective components of the overall system across all phases of the work. Below we highlight the key approaches and results for each team during each major period of research, specifically, the BP, OP1 and OP2. Where possible, we have also identified the relevant program phase in the more detailed technical descriptions that begin in Section 3.0, Technical Approaches.

2.1 ASR

The principal function of the SCRIPTS ASR work was to process audio documents in order to make them sufficiently usable for MT and CLIR. In both the BP and OP1, the acoustic model (AM) parameters were augmented with multilingual bottleneck features, which are discussed in detail in Section 3.1.2, Multilingual AMs. A main distinction between the BP and OP1 for this work was the use of large quantities of untranscribed audio that had been scraped from the web. During OP2, more advanced approaches for using this data, in the form of the lattice-based semi-supervised training (SST) described in Section 3.1.3, Use of Web-Crawled Data and SST, were refined and deployed. From an integration perspective, the simple confidence scores from BP were extended to use full lattices in OP1. This approach was further extended in OP2 to include phrase-based lattice search.

2.2 MT

Each period increased adaptation of MT to program goals and integration with surrounding systems. In the BP, we developed MT systems specifically adapted to the LRL context of MATERIAL via back translation and other features. In OP1, we experimented with forms of *n*-best output, as detailed in Section 3.2.13, *N*-best Translation, to improve indexing and adapt to speech input. Because the languages in OP1 were also higher resource, we were able to improve the system by exploiting changes in neural machine translation (NMT) architectures. In OP2, we found that adding query-driven translation improved acceptance of summaries. At the same time, we found that Kazakh presented an interesting multilingual challenge, given that Russian- Kazakh data is more readily available than Kazakh-English data.

2.3 Data and Language Identification

Training SCRIPTS' tasks on the LRL speech and text corpora for MATERIAL required large amounts of LRL data in both formats, including transcribed speech corpora for ASR, parallel corpora for MT and very large amounts of text data for every other component. During the BP, our group collected very large amounts of text and speech data from the web in three LRLs for this purpose: Swahili, Tagalog and Somali. We also developed new methods of language identification to filter both the data we collected and the language packs provided by IARPA. All of the corpora and tools we collected and created were across all teams to support speech recognition, morphological analysis, language identification, MT, information retrieval (IR), and Summarization. This work concluded following the BP.

2.4 Text Processing

During the BP, we developed *MorphAGram*, a framework for the unsupervised morphological segmentation of a diverse set of languages, including the automatic tailoring of grammars for unseen languages. During OP1 and OP2, we enhanced MorphAGram to allow for the incorporation of linguistic priors, in the form of either grammar definition or linguist-provided affixes.

MorphAGram is described in detail in Section 3.4, Text Processing. During OP1, we also developed unsupervised part-of-speech (POS) taggers using cross-lingual projection using both an averaged perceptron model and a neural model. In OP2, this POS tagger was enhanced through the addition of two key features. First, we developed support for learning from multiple source languages. Second, we made it possible for the system to use both words stems and morphemes as the unit of abstraction during the alignment process.

2.5 CLIR

In the BP, we developed the core capabilities for CLIR, including query analysis, ranking using both MT for 1-best translation and Probabilistic Structured Queries (PSQ) for *n*-best translation, rank-based late fusion for combining system results, cutoff tuning, and formative evaluation. In OP1, we extended our *n*-best approaches both to *n*-best ASR using Keyword Spotting (KWS) and through *n*-best SMT and *n*-best NMT and a Neural Network Lexical Translation Model (NNLTM). In that period, we also built a broader range of rankers, we developed techniques for score-based late fusion and query-specific cutoff selection and we began experimenting with short duration evaluation sprints. Principal research foci in OP2 included development of neural ranking and document expansion techniques and investigation of architectures for coupling early fusion, late fusion, and neural reranking.

2.6 Summarization

In the BP, the novel setting and absence of training data meant that our summarization work focused on using unsupervised methods derived from word embeddings to select relevant sentences from documents. During OP1, we began to overcome the training data limitation by generating synthetic training data and using it to train supervised query-sentence relevance models to perform the same task. During OP2, we augmented our approaches using retranslation to help ensure that in cases where we are highly confident that a given document is relevant to a query, the query itself appears in the document verbatim. This reflected findings from our early work in BP that human end users often failed to mark a summary as relevant unless the precise query terms were present - and visually highlighted - within the summary text presented.

2.7 Integration

During the BP, the decision was made to structure the integration pipeline as a fully configurable system whose parameters could be manipulated solely through options set in an external configuration file. This design decision made it possible to script the execution of a large number of experiments, run small modifications of the experiments (e.g., different translation types), and to trivially apply a given configuration to another dataset. The use of a meaningful directory structure and unified naming convention also facilitated the process of selecting system component versions to test. While the overall pipeline was constant across all three periods during OP2, we added support for processing multiple query sets as well as the reranker component. We also began Dockerization of the executive component during OP2 in order to facilitate the transmission and reuse of the SCRIPTS pipeline and its individual elements.

3.0 TECHNICAL APPROACHES

3.1 Speech Processing

This section describes the approaches used by the SCRIPTS team to process the audio documents so that they can be processed for MT and CLIR.

3.1.1 Baseline System and Diarization

The baseline ASR systems used by both EDIN and CUED teams were hybrid hidden Markov/ Deep Neural Network (HMM-DNN) models built using the Kaldi Toolkit [3]. All AMs developed by both teams were trained on language specific (LS) data following a standard lattice free maximum mutual information (LF-MMI) recipe [4] with a convolutional neural network/time-delay neural network factorization (CNN-TDNNF) architecture that combined 40-dimensional melfrequency cepstral coefficient (MFCC) features with 100-dimensional I-vectors for speaker adaptation [5], [6]. This architecture was used for both narrow-band (NB) and wide-band (WD) data, as described in Section 3.1.3, Use of Web-Crawled Data and SST.

The AMs were trained with natural gradient [7], and with Dropout [8] and SpecAugment [9] for regularization. We also investigated how using Grapheme-based AMs (models mapping from acoustic data to graphemes and utilizing a grapheme-based lexicon to arrive at words) impacted performance as compared to traditional Phoneme models as these approaches have been successfully applied in the Babel program. For each language, we also experimented with adding 100 hours of English data to the language-specific training data to increase the training data size and to improve the model's generalization. These models relied on CNN-TDNNF networks with shared hidden layers but language-specific output layers - and thus language- specific phonesets - and were trained in a multi-task fashion.

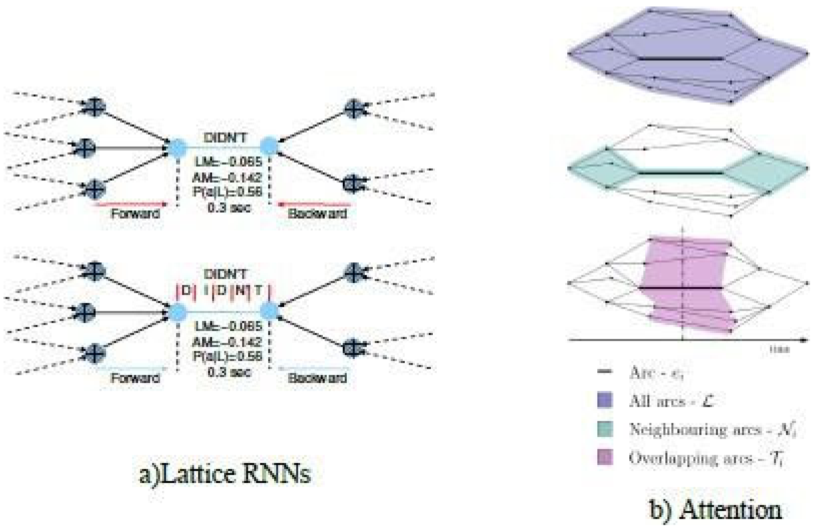
For each language model (LM), we also experimented with data from the Commoncrawl dataset[1] in addition to the Build data. Using the SRI language modeling toolkit (SRILM) [10], we trained three-gram LMs with Kneser-Ney smoothing and a maximum vocabulary size of 300k words, along with a pruning threshold of 1e-9. We also trained recursive neural network LMs (RNN-LM) for second pass rescoring [11].

For both the BP and OP1, the AM parameters already described were augmented with multilingual bottleneck features, are discussed in detail in Section 3.1.2, Multilingual AMs. These hybrid systems were constructed using *n*-gram and RNN-based LMs which were combined using standard linear interpolation, though the features used varied from language to language.

At the start of OP2, the baseline hybrid systems were compared with so called *end-to-end* (E2E) systems, in this case the encode-decoder attention architecture from the Efficient Spatial Pyramid Network (ESPnet) toolkit [12].

As these E2E systems are known to be "data hungry," they were only evaluated on WB data using the web crawled data, as described in Section 3.1.3.

[1] http://data.statmt.org/ngrams/raw/



a)Lattice RNNs

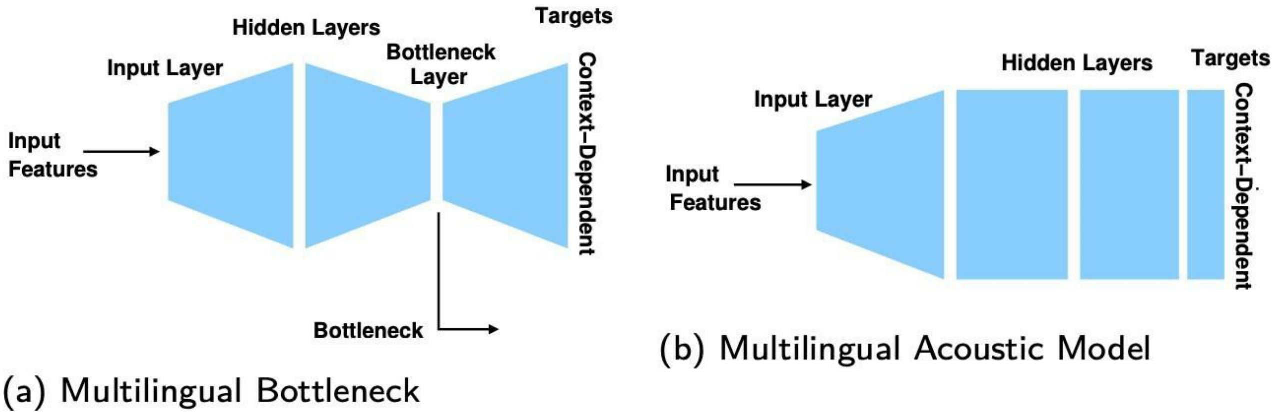
b) Attention

Figure 1. ASR Arc-level Confidence Score Approaches.

Obtaining accurate confidence scores is a key challenge of using speech recognition systems in both low resource scenarios or across mismatched topic domains. In general, the baseline approach for obtaining confidence scores is to combine the arc posteriors from a word lattice with the posteriors from arcs associated with the same word and time instance together, in order to generate a word level posterior. Because this approach often overestimates word-level confidence scores, in Kaldi-based systems they are typically calibrated using decision trees. During BP and OP1, we investigated alternative approaches for generating more accurate confidence scores based on deep-learning (DL) approaches [13], [14], [15], specifically lattice RNNs (illustrated in Figure 1 (a)) and attention mechanisms (Figure 1(b)). Though not used for the final CLIR systems, the impact of these systems in terms of confidence scores is discussed in Section 4.1.1, Baseline Systems and Diarization.

3.1.2 Multilingual AMs

The MATERIAL program's focus on LRL scenarios meant that the quantity of transcribed data available for training the ASR system was sharply limited. One method of addressing this problem was to use a multilingual configuration that incorporates training data from other languages; two standard approaches for accomplishing this are shown in Figure 2. In the multi-lingual bottleneck approach shown in Figure 2 (a), key features are extracted from multiple higher resource languages and are then used to do feature extraction for the target language. The second approach is to use a common set of layers for all languages except for the final classification layer, where language-specific "hats" are applied, as shown in Figure 2(b). For the target language, this means that a new "hat" is only required in the final layer, while the underlying layers can remain the same or optionally be fine-tuned to the target language.



(a) Multilingual Bottleneck

(b) Multilingual Acoustic Model

Figure 2. Multilingual Model Configuration: (a) Bottle-Neck Features; (b) "Hat-Swapping."

3.1.3 Use of Web-Crawled Data and SST

While the MATERIAL program provided only conversational telephone speech (CTS) style data for training the ASR systems, the target systems needed to operate in both CTS and WB signal environments, including news broadcasts (NB) and topical broadcasts (TB). To address this problem, we scraped audio data from the web, focusing on content we believed to be closely related to the target domain. In the BP, we explored using the limited transcriptions available for this web-scraped data to train our model, but found they were insufficient for building high performance ASR systems. In the subsequent OP1 and OP2 periods, we instead trained the ASR systems by using the limited transcription data as "seed" models in order to generate transcriptions for the web-scraped audio data. Accurate transcriptions were then selected to train a new system in the target domain.

Note that in cases where only web-scraped data is available for a particular domain, there may not be any accurately transcribed data in the target domain to support system development and hyper-parameter tuning. To address this problem during BP and OP1, we examined the possibility of developing an "in the wild" ASR [16]. Rather than using word error rate (WER) based on accurate transcriptions to tune model configurations, this approach relies on systems developed by maximizing (approximately calibrated) confidence scores. This makes it possible to tune and configure ASR systems even on development sets that lack transcriptions entirely.

Figure 3 shows the cumulative plots of confidence scores for the web-crawled audio data for Swahili and Somali. At CUED, a simple data selection scheme was adopted in which only data above a particular confidence threshold was used to train the ASR system. However, as one of the differences between the transcribed data and the target domain was the bandwidth (i.e., 8KHz for CTS, 16KHz for WB), once the WB training data had been selected new models were trained from scratch. This allowed features based on the full WB audio bandwidth to be used and was found to be more effective than using NB features combined with the transcribed data. Depending on the available time, multiple iterations of data selection can be performed. Given the quantity of data that can be crawled from the web, iterative selection can be run on different batches of web-crawled data rather than iteratively improving the transcription on the same block of web-crawled data. No direct comparisons ran for this and the implementation per language depended on the quantity of web-crawled data available for system build.

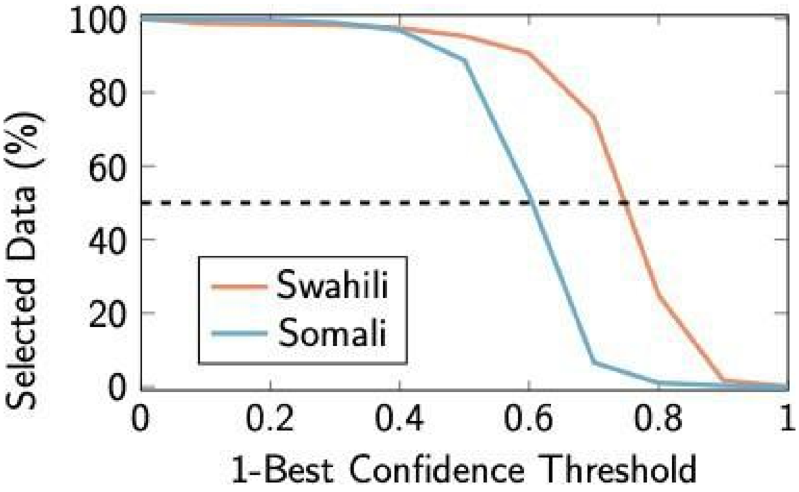


Figure 3. Confidence-Based Web Data Selection.

To build upon the seed/baseline models described in Section 3.1.1, the EDIN pipeline utilized lattice-based SST to improve the AM quality/robustness and directly tackle the domain-gap/ mismatch between the training and analysis/test data. Specifically, we used the baseline systems as seed models to provide pseudo-labels for additional, web-crawled, un-transcribed data, which we expect would exemplify a far broader range of domains than the telephone speech of the provided training data. In contrast to the CUED approach, we used pseudo labels in the form of lattices in order to explicitly model the uncertainty of the seed model.

For each language, the EDIN system obtained this additional data by scraping YouTube videos, using the most common trigrams from the relevant LM as queries. Because this data is likely to be noisy - containing audio other than speech, and languages other than the target - we employed filtering at the video level. We first decoded the data using the seed AM and LM and discarded videos with resulting mean confidence falling below 0.7. We also discarded videos where average speaking rate identified by voice activity detection (VAD) fell below 1.25 words per second; this helped to filter out non-speech data such as music videos, where confidence levels could be erroneously high. The thresholds for both mean confidence and speaking rate were developed by comparing the distributions of these parameters' values for the web-scraped data against their value for our development set - which we knew to be of good quality - and with an out-of language (OOL) dataset. These results are shown in Figure 4

In a standard lattice-based SST approach using LF-MMI [17], all unlabeled data is decoded by the seed model in a single round, with one subsequent SST model then trained on this newly transcribed data. In our system, we instead employed an incremental Semi-Supervised Training (iSST) setup in which we split the scraped data into *n*-equally sized chunks and decoded them iteratively. In practice, this means that a given chunk is processed using a model trained only on the previous chunk, so that the pseudo labels obtained when decoding the i-th chunk are the only ones used to train the model that will process the i+1-th chunk. Unlike other iSST approaches (e.g., [18]), this means we never trained twice on the same chunk, which can lead to over-fitting.

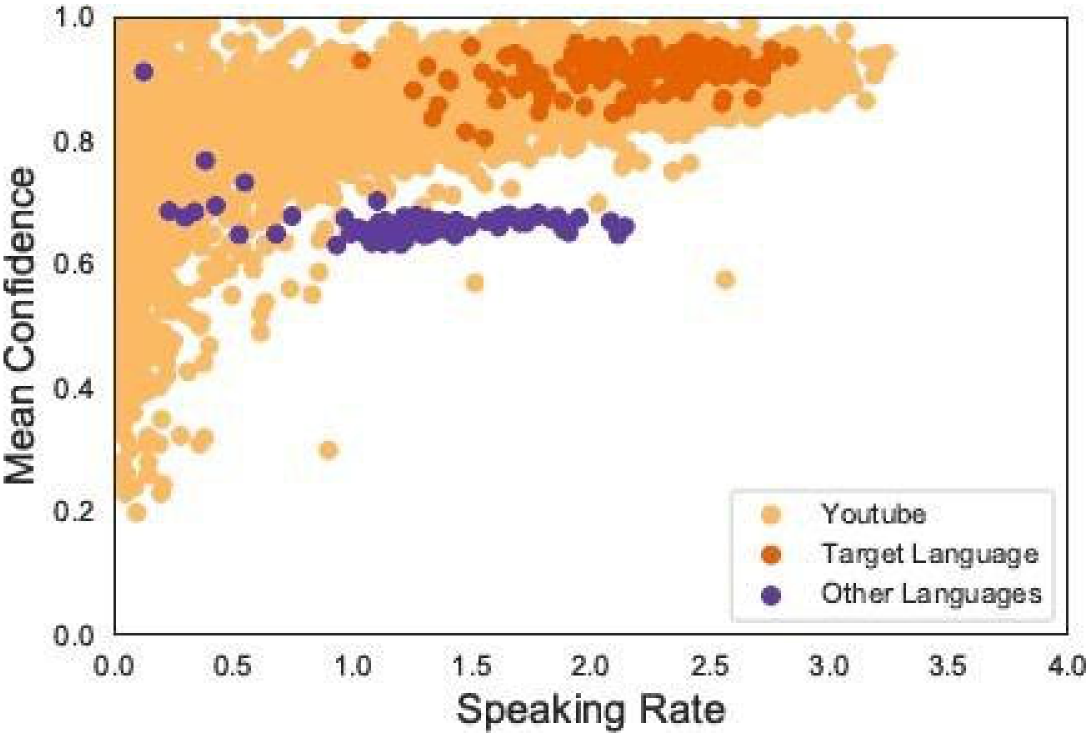


Figure 4. Plotting Speaking Rate Against Mean Lattice Confidence for each Tagalog Utterance in: Raw Tagalog YouTube Data; Tagalog Analysis Dataset ('Target Language'); and OOL Data ('Other Languages').

To reconcile the newly scraped wide-band (16kHz) data and the NB (8kHz) telephone speech that had been used to train the seed models, we were required to down sample the first chunk of web-scraped data before the initial decoding (pseudo label/lattice generation) process. Facilitated by our incremental training setup, we subsequently opted to train all the remaining SST models using the complete WB features, which required using a randomly initialized seed model. An exponential decay training schedule [19] was also used for the continued training. Though this approach meant that we couldn't incorporate any of the initial supervised data into our final models, we found that with correct tuning of our lattice-based SST recipe, trading incorporation of initial data for the ability to use the more informative WB features, was beneficial [20].

We also chose not to apply traditional confidence filtering [21], [22], [19] to the obtained pseudo labels before retraining, as we observed that while it can help to alleviate error propagation, such confidence filtering also filters out the most difficult - and thus useful - training examples derived from the web-scraped data. Instead, we strove to maximize the value of our SST data while still minimizing impact of erroneous labels through three intersecting approaches. Our subsequent investigations [23] revealed that SST for ASR works most effectively when the LM can be relied upon to guide and reduce the frame-level labelling uncertainty of the initial model by providing additional, external, reliable information to the system at the utterance-level. To achieve this, it is necessary to ensure the LM used during decoding was of highest quality possible. We relied on the LF-MMI training criterion to correctly calibrate the confidence levels between word-based lattices from decoding with the seed model, and state-based confidences from the seed AM.

3.1.4 Integration with Downstream Tasks

The ASR work described above was incorporated into the SCRIPTS project at a variety of levels, as described below:

- **Audio Document Language Verification (CUED)**: For the BP, performers were required to appropriately handle audio documents that were not in the target language of interest. Leveraging the confidence work described in Section 3.1.1, the SCRIPTS team was able to adopt a simple ASR confidence-based approach. Word-level confidence scores were computed based on ASR lattices and decision tree normalization to calibrate scores as discussed in Section 3.1.1. These word-level confidence scores were then averaged for the complete documents to yield an overall document-level confidence score.
- **Integration with MT (EDIN):** Our systems integrated with MT at the one-best hypothesis level, optionally after CUED/EDIN system combination. For this, ASR output was acoustically segmented into sentence-like units with a maximum length of 40 words in order to optimize MT performance, though we elected to train the ASR-specific MT systems on text with casing and punctuation removed. While we made a number of efforts to integrate our work more closely with MT - for example, through the use of shared continuous representations and joint optimization - there were several obstacles to these efforts. Specifically, we were hampered by the lack of any translated speech data in the MATERIAL program, without which E2E training is very challenging; we also lacked a common DL toolkit between the two disciplines. We are thus unable to report results on this aspect of the research.
- **Integration with CLIR (CUED)**: In the BP, the integration between ASR and CLIR followed the same process that was used for text-based CLIR. ASR was used to generate text, the one best output, and the text used for CLIR. The only modification to this process was the use of ASR confidence scores, as discussed in Section 3.1.1. During OP1, this was approach was extended to handle word arcs within a lattice, and in OP2 we further extended this to handle phrases. The general structure for this work is shown in Figure 5.

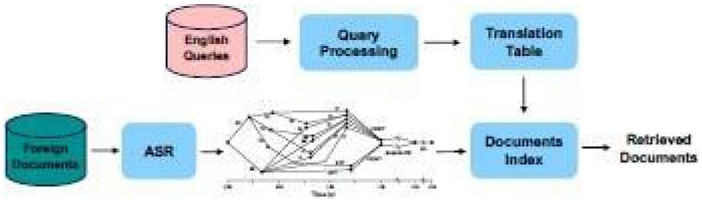


Figure 5. Confidence-Based Language Verification.

To make use of the lattices for CLIR, the HMM query generation model was used. Here the probability of generating query $\boldsymbol{q}^e$ from audio document $\boldsymbol{s}^f$ can be expressed as

$$P(\boldsymbol{q}^{\mathrm{e}}|\boldsymbol{s}^{\mathrm{f}}) = \prod_{\boldsymbol{p}^{\mathrm{e}} \in \boldsymbol{q}^{\mathrm{e}}} \Big[(1-\alpha)P(\boldsymbol{p}^{\mathrm{e}}|\boldsymbol{s}^{\mathrm{f}}) + \alpha P(\boldsymbol{p}^{\mathrm{e}}|\boldsymbol{g}^{\mathrm{e}})\Big] \tag{1}$$

$$P(\boldsymbol{p}^{\mathrm{e}}|\boldsymbol{s}^{\mathrm{f}}) = \sum_{\boldsymbol{p}^{\mathrm{f}} \in \mathcal{L}^{\mathrm{f}}} P(\boldsymbol{p}^{\mathrm{e}}|\boldsymbol{p}^{\mathrm{f}}) P(\boldsymbol{p}^{\mathrm{f}}|\mathcal{L}^{\mathrm{f}})^{\epsilon} \tag{2}$$

where g^{e} is a general English LM, and $calL^{f}$ is the (foreign) word lattice from the ASR decoding of the foreign audio document.

Handling word or phrase-based CLIR differs in the decomposition for $\boldsymbol{p}^{\mathrm{e}} \in \boldsymbol{q}^{\mathrm{e}}$. Specifically, word-based approaches simply split the processed query into individual words and the probabilities of the individual words are then combined to form the probability of the phrase.

Alternatively, the phrase can be partitioned into all possible phrases, and the probabilities of the phrases are then combined. In practice, rather than considering all possible phrases, we applied a simple deterministic process in which the English search phrase was broken down by repeatedly splitting the longest phrases in the phrase translation table P (p^{e} p^{f}). Though finding all phrases in the lattice that occur with a minimum probability n in the translation table increased the size of the document index, the size of the increase was acceptable and made it possible to generate a single index that efficiently supports both word and phrase search.

- Metadata (EDIN): We investigated approaches to automatically predicting speaker metadata. While this could conceivably improve CLIR metrics in some scenarios, it would be difficult to optimize given the lack of speaker metadata in the Analysis and Development sets. However, the ability to directly filter results according to speaker metadata could be useful in a practical system.

We developed a framework for predicting speaker age and nationality from English language speech using only web-scraped data with noisy metadata labels [24]. We used a multi-task learning framework, which enabled training on multiple data sources, each with possibly disjoint sets of speaker metadata. We found that the use of a shared representation for all metadata prediction tasks improved the performance of speaker diarization and verification. Work to investigate the potential to use these models in a cross-lingual setting for application on LRLs is at a preliminary stage.

3.2 MT

The key steps in developing MT systems include the data selection and cleaning process (discussed in Section 3.2.1), applying SMT by generating translations through the use of statistical models that analyze features of bilingual texts; and a variety of E2E NMT approaches. In the following section, we discuss the MT portion of the SCRIPTS project, focusing on the approaches tested or used in developing our final system.

3.2.1 Data Selection and Cleaning

For each language in the MATERIAL BUILD corpus, both teams followed a data augmentation, selection and cleaning process that starts by incorporating both parallel and English monolingual corpora from Spinn3r [25] and Newscrawl.[2]

Additional training data - both monolingual and parallel - was gathered from the web. Building upon work from the European Union (EU) - funded ParaCrawl project,[3] we trained a first pass MT system on MATERIAL and other readily available corpora - such as the Open Parallel Corpus (OPUS) - then translated all monolingual text in languages of interest to English.

Documents were matched in English using term frequency-inverse document frequency (TF-IDF) then translated sentences were extracted using bleu align [26] and cleaned using BiCleaner [27].

These data sources are then cleaned and regularized using an initial data preprocessing pipeline outlined below:

- Remove non-printing characters (e.g., page breaks) using the Moses[4] scripts.
- Normalize punctuation using the Moses scripts.
- Remove duplicate sentence pairs.
- Filter out sentence pairs with length ratio larger than three.
- Normalize characters for each language using the normalization tool described in Section 3.4.3.
- Tokenize and true case using the Moses scripts.
- Apply byte-pair encoding (BPE) segmentation for neural translation [28] models—which combines commonly co-located character sequences - using the subword-nmt toolkit.[5]

In OP1 and OP2, we add a filtering step using the language identification toolkit, and switch from subword-nmt[5] to fastBPE[6] to obtain faster BPE segmentation. We also incorporate new versions of the character normalization tool tailored to the character set of each of the new languages in each phase of the program. Table 1 shows the data statistics for each language pair after preprocessing.

[2] http://data.statmt.org/news-crawl/en/

[3] The goal of ParaCrawl https://paracrawl.eu/ was to create parallel corpora for official European Union languages, including Bulgarian and Lithuanian. As a result, corpora for these languages had already been created at the time of their announcement.

[4] http://www.statmt.org/moses/

[5] https://github.com/rsennrich/subword-nmt

[6] https://github.com/glample/fastBPE

Table 1. Number of Sentences (K) in the Parallel Corpora after Preprocessing.

	Tagalog	Swahili	Somali	Lithuanian	Bulgarian	Pashto	Farsi	Kazakh	Georgian
MATERIAL BUILD	51	24	24	42	41	44	34	9	4
GlobalVoices	1	28	0	0	0	0	8	0	0
JW300	0	0	0	0	0	0	269	0	419
Panlex, Corp.dict	107	190	0	0	0	7	0	12	147
LORELEI	33	0	46	0	0	0	52	0	0
OpenSubtitles, TED	0	0	0	0	0	0	4748	11	191
Europarl	0	0	0	613	394	0	0	0	0
Rapid2016	0	0	0	211	198	0	0	0	0
Paracrawl	21	0	0	816	1015	84	177	0	0
Commoncrawl	18	9	0	0	0	0	0	0	0
BBNBitext	0	0	0	0	0	223			
Wikipedia, Wikimedia	0	0	0	0	0	0	77	0	15
WikiMatrix	0	0	0	157	358	0	0	13	10
WMT-News	0	0	0	0	0	0	0	4	0
Kazakhcrawl, Kazakhtv	0	0	0	0	0	0	0	515	0
MultiCCAligned	0	0	0	0	0	0	0	423	1029
OPUS-tech	0	0	0	0	0	19	54	35	46
OPUS-others	0	0	0	0	0	0	2581	5	6
Total	232	252	70	1839	2005	377	8019	1028	1869

3.2.2 SMT

The SMT models are all phrase-based MT models, based on the Moses SMT toolkit [29]. Given an input sentence *f*, translation hypotheses *e* are scored using a standard log linear model as follows:

$$e^* = \arg\max p(e|f) \tag{3}$$

$$= \arg\max_e \left\{ \exp\left(\sum_i \lambda_i h_i(e, f) \right) \right\} \tag{4}$$

where h_i are feature functions and λ_i are feature weights optimized on the development set using the minimum error rate training (MERT) algorithm [30].

Our phrase-based system uses a standard best practices combination of the phrase translation, lexical reordering, and LM feature functions - which capture the relationship between the source and target - as outlined below:

- phrase translation model (four features)
- distance-based reordering model (one feature)
- lexicalized reordering model (six features)
- LM (two features when translating into English,
  one when translating out of English)
- word penalty (one feature)
- distortion (one feature)
- unknown word penalty (one feature)
- phrase penalty (one feature)

Phrases and lexical reordering features are extracted from the parallel training data, by extracting phrase pairs that are consistent with an automatic word alignment of the corpus. Word alignments in each translation direction are obtained using a multi-threaded version of GIZA++ [31]. The bi-directional alignments are combined starting from the intersection of the two alignments, which are then combined with further alignment points according to a *grow-diag-final-and* heuristic.[7] We then use the *msd-bidirectional-fe* configuration[8] for the reordering model. This approach relies on both backward and forward models and is conditioned on both the source and target languages. It also considers the monotone, swap, discontinuous orientations.

LM features are based on five-gram LM with Modified Kneser-Ney smoothing, as implemented in lmplz [32]. All systems include an LM trained on the target side of the parallel corpus. For systems with English as the target language, we train a second five-gram LM on the 1.3M sentences from the Spinn3r corpus.

The decoder performs translation using a beam search [33], [34], and optionally outputs sentence-level *n*-best hypotheses.

3.2.3 UMD NMT

All the UMD NMT systems are implemented using the Sockeye toolkit [35]. We train bidirectional translation systems, in which a single model is trained for both directions of a language pair [36]. During training and inference, we also add an artificial token to the beginning of each source sentence to mark the desired target language. This allows us to back-translate source or target monolingual data without requiring an auxiliary model.

In the BP, we use a bidirectional Long Short-Term Memory (LSTM) architecture with embed- ding size of 512 for the LRL Swahili and Somali. In subsequent periods, we switched to the Transformer architecture to improve translation quality for higher resource languages such as Lithuanian and Bulgarian. Specifically, we use the base Transformer architecture [37] with six layers for both encoder and decoder, an embedding size of 512, a feed-forward network size of 2048, eight attention heads, and residual connections. We adopt layer normalization [38] and label smoothing [39] to avoid overfitting. We also filter out sentences with a length of more than 80. Using an Adam optimizer [40] and a batch size of 2,048 words, we checkpoint models every 1,000 updates. This configuration yields an initial learning rate of 0.0002, which is reduced by 30% after the perplexity on the validation set stops decreasing for five checkpoints. Training stops after 20 checkpoints without improvement. During inference, the beam size is set to five. This reflects the process used in the final evaluation system.

3.2.4 Other Experimental Approaches at UMD

In OP2, we also experimented with a range of other NMT architectures that were not used in the final systems. Specifically, we introduced an Edit-Based Transformer with Repositioning (EDITOR) [41], which makes sequence generation flexible by seamlessly allowing users to specify preferences in output lexical choice. Building on recent models for non-autoregressive sequence generation [42], EDITOR generates new sequences by iteratively editing hypotheses.

By using a novel reposition operation designed to disentangle lexical choice from word positioning decisions, it enables efficient oracles for imitation learning and parallel edits at decoding time.

[7] http://www.statmt.org/moses/?n=FactoredTraining.AlignWords
[8] http://www.statmt.org/moses/?n=FactoredTraining.BuildReorderingModel

3.2.5 EDIN NMT

The EDIN team focused on developing NMT systems based on the Marian [43] MT toolkit. Systems were generally trained for one direction at a time; the exception was Kazakh, where we experimented with multilingual Russian, Kazakh, and English systems. Comparing the Transformer architecture [37] with LSTM-based models revealed that Transformer models produced notably better results, so all evaluation systems and docker containers are based on the Transformer architecture. We used six decoder layers, six encoder layers, and eight of each type of head.

For low resource MT, we follow our research group's guidance [44] in setting a small vocabulary, avoiding overfitting through early-stopping on a validation set - even where this meant sacrificing some of the data - and by applying the methods described in Section 3.2.8, Shared NMT Adaptations.

We applied multi-agent dual learning [45] over several rounds to make thorough use of available monolingual data, as languages like Pashto had little monolingual text available, even on the web. Similarly, we performed two rounds of back-translation [46] for Pashto and other particularly LRL.

3.2.6 Query-Guided Translation

User ratings indicate a preference for summaries that contain query words directly, rather than those that contain synonyms of query words. As such, we created a query-guided system takes the input sentence along with a query (token *n*-gram) in target language and generates a fluent translation with the query in it. We accomplish this via a simple data manipulation at the source side of the training data. Specifically, for each training example we randomly choose an *n*-gram from the target side and append it on the source side with a special token indicating whether the input has a query in it. In our experiments, we restrict the system to using maximum of three tokens (0 token is equivalent to no-query) in a query, however, the model can generate translations with both longer queries and no queries. Because using this method means that query words will appear in the output even if the sentence is irrelevant, it was the responsibility of CLIR (see Section 3.5, CLIR) to determine when to force their presence.

3.2.7 Fine Tuning Persian→English for Tweet Translation

As we had only a small volume of parallel tweets for Persian-English, we relied mainly on back translation and the pre/post-processing of the source/target tweets for adapting the system to tweet translation. In the pre-processing stage, we first replaced the Twitter user names (handles) and Uniform Resource Locators (URL) with placeholders (e.g., HANDLE0, HANDLE1, so on, and URL0, URL1, so on). In the post-processing stage, we once again substituted the place- holders with the original values.

For fine tuning, we followed the pre-processing steps used in the baseline system on top of Twitter-specific pre-processing. To fine tune a model for tweets, we first initialized a baseline model and then we fine-tuned it in both directions to forward/back-translate the original tweets and use them as additional parallel tweets. We subsequently performed an additional round of fine tuning, and then shipped the ensemble of four such fine-tuned models.

3.2.8 Shared NMT Adaptations

In addition to the methods listed individually above, UMD and EDIN shared techniques to adapt to the MATERIAL setting and improve quality, especially in low resource settings. Both systems used dropout of 0.1 to the encoder and decoder states. We also tie the output weight matrix with the source and target embeddings [47], which provides more opportunities to train word embeddings and increases copy performance.

3.2.9 Improving NMT via Back Translation

We further improve NMT performance through back translation [46]. UMD's theoretical and empirical work [48] shows that iterative back- translation is more effective than dual learning

[49] despite its relative simplicity. For example, for Kazakh to English translation, we first train a bidirectional NMT model on the parallel data, and then use it to back translate the English monolingual texts into Kazakh, thereby generating a pseudo-parallel corpus. Finally, we train a new NMT model on the 1:1 combination of the parallel and pseudo-parallel data.

3.2.10 Improving NMT via Pivot-based Data Augmentation

For LRLs that are closely related to high/medium resource languages, we can augment the training data via pivot translation. For instance, Russian is a high resource language related to Kazakh. Thus, we first train a Russian-English NMT model using the Workshop on Machine Translation (WMT) 2019 [50] parallel data, and then use it to translate the Russian side of the Kazakh-Russian parallel data into a pseudo Kazakh-English corpus.

3.2.11 Stemmed English Systems

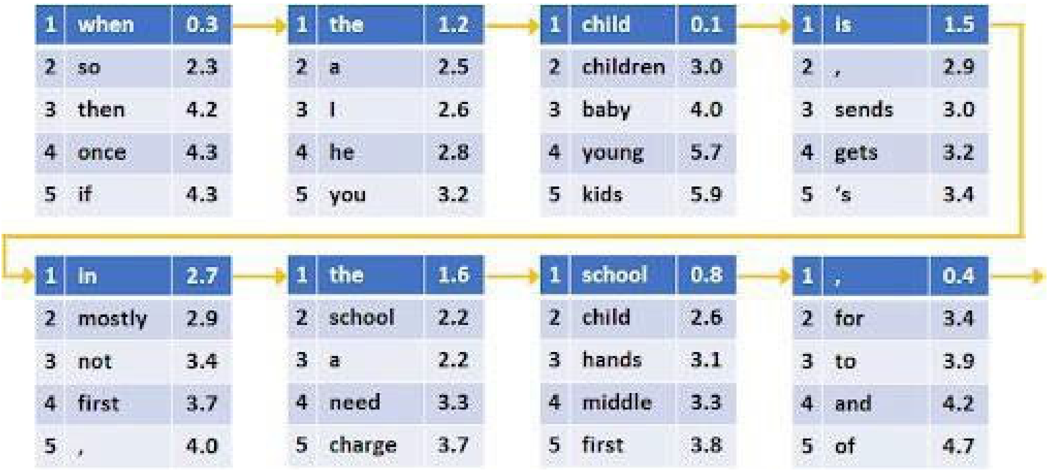
To improve interaction with the IR system, we also provide NMT systems trained on a version of the training data where the English side is stemmed with the Porter stemmer from the Natural Language Toolkit (NLTK).

3.2.12 Translating Speech

To improve NMT for the ASR outputs, we train an additional NMT system for each translation direction on data that we process to be more "speech like" by lower casing all text and removing punctuation.

3.2.13 N-best Translation

In addition to generating 1-best translations using beam search, we also provide the IR system with token-level n-best translation options. During beam search for the 1-best translation output, we record the top *n* translation candidates at each time step, as well as their probability according to the NMT model (see the example in Figure 6). This is different from the typical *n*-best hypotheses protocol provided by the toolkit which operates on the sentence level. By doing this, the resulting CLIR system is more robust to severe errors made by the NMT model, as the IR model can discount the words or options for which the NMT model is low confidence. This also promotes higher recall by providing IR with a larger set of word translations.



1	when	0.3	1	the	1.2	1	child	0.1	1	is	1.5
2	so	2.3	2	a	2.5	2	children	3.0	2	,	2.9
3	then	4.2	3	I	2.6	3	baby	4.0	3	sends	3.0
4	once	4.3	4	he	2.8	4	young	5.7	4	gets	3.2
5	if	4.3	5	you	3.2	5	kids	5.9	5	's	3.4

1	in	2.7	1	the	1.6	1	school	0.8	1	,	0.4
2	mostly	2.9	2	school	2.2	2	child	2.6	2	for	3.4
3	not	3.4	3	a	2.2	3	hands	3.1	3	to	3.9
4	first	3.7	4	need	3.3	4	middle	3.3	4	and	4.2
5	,	4.0	5	charge	3.7	5	first	3.8	5	of	4.7

Figure 6. An Example of Token-Level n-best Translations with Probabilities.

3.2.14 Improving NMT via Ensemble Decoding

For all language pairs, we train four NMT models with different random seeds and use the ensemble model for decoding.

3.3 Data Collection and Language Identification

The goal of the IARPA MATERIAL program was to design and build software systems capable of finding content in text and speech documents relevant to an input query consisting of a do-main and an English query string. The system output was an English summary of IR to the specified query and domain. However, the text and the speech documents from which the information was to be retrieved were in a variety of LRLs. Training SCRIPTS' tasks on LRL speech and text corpora required large amounts of LRL data in both formats, including transcribed speech corpora for speech processing (ASR), parallel corpora for MT and very large amounts of text data for every other component.

However, the small size of the official LRL language packs released by IARPA for the project meant that procuring new data sources to train system modules on was essential. Our group collected very large amounts of text and speech data from the web in three LRLs for this purpose: Swahili, Tagalog and Somali. We also developed new methods of language identification to filter both the data we collected and the language packs provided by IARPA.

To do this collection, we developed a number of new data collection methods, including a pipeline to download audio, titles and captions from YouTube videos using keywords extracted from the target languages. We also built a tool to scrape top Swahili and Tagalog news texts using the Python programming language and the library BeautifulSoup.[9] We also leveraged Babler [51], a tool created in our Speech Laboratory, which uses Microsoft Bing Search to query blog posts and first segment them into sentences, then normalize and tag them with a language identifier.

We cleaned the data by developing new tools and methods for both language identification and POS. We normalized the data by removing all non-language characters, tokenizing sentences

[9] https://pypi.org/project/beautifulsoup4/

using the NLTK Punkt tokenizer [52], and removing remaining punctuation. We also developed language identification tools at the document and sentence level using the majority vote of the confidence scores produced by three freely available tools: TextCat,[10] Google's Compact Language Detector (CLD),[11] and LingPipe.[12] We then used *code-switching* techniques to identify switches among two or more languages in text or speech conversation. These were applied to Tagalog, Swahili and English corpora to create *anchor words* that are unique to a single language among a large pool of 134 other languages obtained from a Wikipedia corpus that we curated for this work.

All of the corpora and tools we collected and created were across all teams to support speech recognition, morphological analysis, language identification, MT, information retrieval (IR), and summarization.

3.4 Text Processing

The focus of the text processing portion of the research is on developing morphological analyzers. In this project, we have developed two main capabilities: 1) morphological segmentation; 2) and POS tagging. In addition, we provided a text normalization component that does text cleanup, transliteration and a number of other operations to control numbers, punctuation marks, repetitions and foreign text.

3.4.1 MorphAGram: Unsupervised Morphological Segmentation

Throughout the course of the MATERIAL program, we developed MorphAGram, a state-of-the-art (SOTA) publicly available framework for unsupervised and minimally supervised morphological segmentation that is based on Adaptor Grammars (AG) [53]. AGs are nonparametric Bayesian models that generalize probabilistic context-free grammars (PCFG). An AG is composed of two main com- ponents: a PCFG and an adaptor. In the case of morphological segmentation, the PCFG is a morphological grammar that specifies word structure, while the adaptor adapts the subtrees and their probabilities to the corpus they are generating and acts as a caching model. The adaptor used in MorphAGram is based on the Pitman-Yor process [54].

In MorphAGram, we define several language-independent grammars and introduce different learning settings that are either unsupervised or minimally supervised. MorphAGram also allows for the automatic tailoring of grammars for unseen languages and for the incorporation of linguistic priors, in the form of either grammar definition or linguist-provided affixes. We describe the different components and modules of the MorphAGram framework below.

3.4.1.1 Grammars

The first step in learning morphological segmentation using AGs is to define the grammars that will be used to model word structure. The definition of a grammar relies on three main dimensions:

- Word Modeling: A word can be modeled as a sequence of generic morphemes/morphs or as a sequence of a prefix, a stem and a suffix, where any nonterminal may be recursively defined to allow for compounding.
- Level of Abstraction: Basic elements can be combined into more complex nonterminals, e.g., Compound, or split into smaller ones, e.g., SubMorph

[10] https://github.com/wikimedia/textcat

[11] https://github.com/google/cld3

[12] http://www.alias-i.com/lingpipe/

- Segmentation Boundaries: This dimension defines the nonterminals that incur splits in the final segmentation output. For example, a word can be segmented into a complex prefix, a stem and a complex suffix (three-way segmentation), e.g., redis+cover+ing and irre+place+ables, or it can be split into a stem and simple affixes (multiway segmentation), e.g., re+dis+cover+ing and ir+re+place+able+s.

Table 2 lists nine grammars and their characteristics. While *Morph+SM* and *PrStSu2a+SM*are baseline grammars first introduced by [55], the remaining grammars are novel derivations that we introduce and include in the *MorphAGram* framework. Choosing the proper grammar could depend on a combination of the target language and the downstream application: certain grammars perform better with certain languages, while specific non-terminals or degrees of granularity in a grammar may affect the induced segmentation in the down-stream application. Though we derived and experimented with grammar variations beyond those outlined in Table 2, we eliminated those that generally do not perform well across our development languages of English, German, Finnish, Estonian, Turkish and Zulu. Through experimentation, we found that the best-on-average grammar across development and test languages was *PrStSu+SM*.

3.4.1.1.1 Word Modeling

With respect to word modeling, all the listed grammars model the word as a sequence of a prefix, a stem and a suffix, except the Morph+SM grammar, in which the word is modeled as a sequence of morphs. In addition, in all the grammars denoted by PrStSu, prefixes and suffixes are recursively defined as a sequence of affix morphs in order to allow for affix compounding.

3.4.1.1.2 Level of Abstraction

The level of abstraction of all the grammars denoted by *Co* involves a high-level nonterminal, Compound that expands to a prefix, a stem or a suffix; those denoted by SM involve a low-level nonterminal, SubMorph, that expands to a sequence of characters. These nonterminals allow prefixes, stems and suffixes to share common information, which is efficient for languages of rich affixation. We use the labels 2a and 2b to denote binary high-level non-terminals that combine stems with suffixes (Stem-Suffix) and prefixes with stems (Prefix-Stem), respectively.

Table 2. Grammar Definitions for Modeling Word Structure. Y=Applicable.

Grammar	Word Modeling	Compound	Morph	SubMorph	Segmentation Boundaries
Morph+SM	Morph		Y	Y	Morph
Simple	Prefix+Stem+Suffix				Prefix-Stem-Suffix
Simple+SM	Prefix+Stem+Suffix			Y	Prefix-Stem-Suffix
PrStSu	Prefix+Stem+Suffix		Y		PrefixMorph-Stem-SuffixMorph
PrStSu+SM	Prefix+Stem+Suffix		Y	Y	PrefixMorph-Stem-SuffixMorph
PrStSu+Co+SM	Prefix+Stem+Suffix	Y	Y	Y	Prefix-Stem-Suffix
PrStSu2a+SM	Prefix+(Stem-Suffix)		Y	Y	PrefixMorph-Stem-SuffixMorph
PrStSu2b+SM	(Prefix-Stem)+Suffix		Y	Y	PrefixMorph-Stem-SuffixMorph
PrStSu2b+Co+SM	(Prefix-Stem)+Suffix	Y	Y	Y	Prefix-Stem-Suffix

In addition to defining the main grammar, each production rule has to be associated with three parameters; θ, *a* and *b*, where θ is the probability of the rule in the generator, while *a* and *b* are the parameters of the Pitman-Yor process [56]. If not otherwise specified, the parameters can either be sampled by the learner or set to default values prior to running it. Specifically, setting *a = 1* means the underlying non-terminal is not adapted and is therefore sampled by the general Pitman-Yor process; setting $a = 0$, meanwhile, indicates that the non-terminal adaptor is a Dirichlet process [57] with the concentration parameter b. When a non-terminal is adapted, any sub-tree that can be generated using the initial rule of that non-terminal is considered a potential rule in the grammar. Otherwise, the non-terminal expands as in a regular PCFG, as outlined in [53] and [58].

3.4.1.2 Training

The two main inputs to the learner are the grammar (including any adaptation information), and the vocabulary of the target language (represented as a list of unique, unsegmented words). If the size of the vocabulary is relatively large (e.g., more than 50,000 words), we recommend providing only the most frequent words in the target language, as the segmentation of the remaining words can be obtained through inductive learning. In the following section, we present the three types of learning settings used to train the model: Standard, Scholar-Seeded and Cascaded.

3.4.1.2.1 Standard Setting

In this setting, we train a morphological-segmentation model using a language-independent grammar. This type of model is not seeded with knowledge about the underlying language nor will it model language-specific characteristics. This setting is typically used when processing an unseen language or a language whose description is inadequate or lacking, as in the case of some low-resource and endangered languages.

3.4.1.2.2 Scholar-Seeded Setting

In this setting, we seed the model with scholarly knowledge compiled from already existing language resources to create a more informed morphological-segmentation model. This scholar-seeding approach leverages the fact that for many languages, relatively extensive descriptions of their morphology - such as context-free listings of affixes - are avail-able through resources such as the Wiktionary[13] and other online grammar references. AGs are a framework that is particularly well suited for applying scholar-seeded knowledge, as AGs take as input hand-crafted grammars. This allows us to insert affixes into these grammars in the positions where the affixes are generated, while still allowing the grammars to generate new affixes in order to compensate for the incomplete listing we expect to find in the scholarly resources. Such affixes can be incorporated as either adapted or unadapted nonterminals. With lower quality affixes in particular, it is advisable to use them as unadapted in order to prevent the sampler from spreading wrong information by generating multiple instances of the corresponding subtrees.

3.4.1.2.3 Cascaded

The Cascaded setting approximates the effect of the Scholar-Seeded setting, but in language-independent manner through the use of self-training. The Cascaded setting relies on two rounds of learning. In the first round, we train a morphological-segmentation model using a high precision grammar as we did in the Standard setting and extract the list of the most common affixes from the segmentation output. Next, we seed the list of extracted affixes into the grammar of interest as unadapted nonterminals, using a similar seeding approach to that used in the Scholar-Seeded setting. We then train another morphological-segmentation model using the augmented grammar in a second round of learning. The base grammar selected in our Cascaded setting is chosen independently of the language in order to derive a language-independent morphological segmentation. We do this by optimizing on precision (rather than high F1-score) so we can be

[13] https://en.wiktionary.org/wiki/Wiktionary:Main_Page

certain of having true affixes in the grammar, rather than having as many affixes as possible (including some that are incorrect). Therefore, we choose the *PrStSu2b+Co+SM* grammar as it achieves the highest average precision when evaluated on our development languages. To determine the optimal number of affixes for the *Cascaded* setting, we also ran experiments in which we extracted and seeded *n* affixes, where *n* {10, 20, 30, 40, 50, 100} leading to the discovery that *n* = 40 yields the best average performance across the development languages - specifically, English, German, Turkish, Finnish, Estonian and Zulu. Accordingly, we set *n = 40* in all our Cascaded setups. Figure 7 illustrates the Cascaded learning setting, where the prefixes re, im and ex and the suffixes er and s are generated from a first round of learning using the *PrStSu2b+Co+SM* grammar and seeded into the *PrStSu+SM* grammar as PrefixMorph and SuffixMorph production rules, respectively, for a second round of learning.

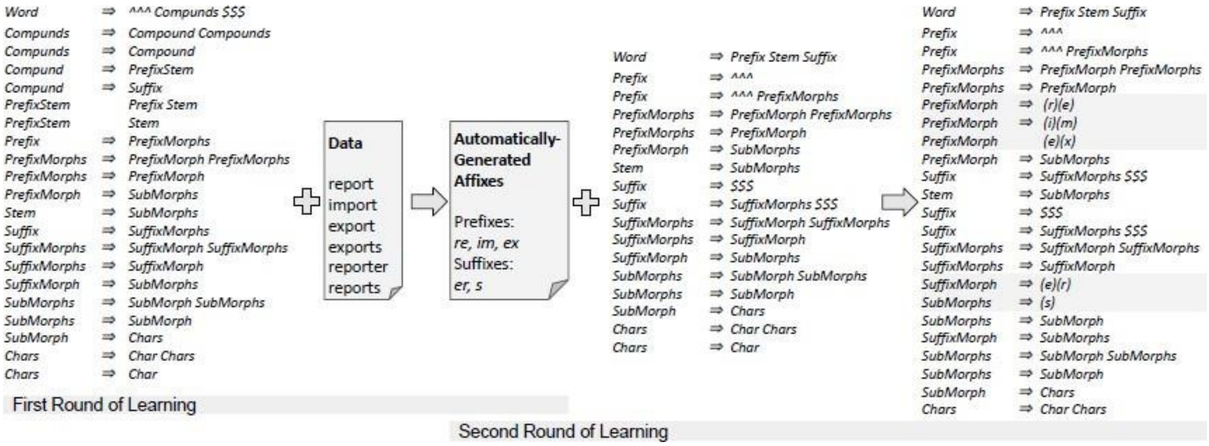


Figure 7. An Example of the Cascaded Setting.

Where some English Affixes are Extracted from an Initial Round of Learning Using the PrStSu2b+Co+SM Grammar and Seeded into the PrStSu+SM Grammar for a Second Round of Learning.

3.4.1.3 Including Linguistic Priors

MorphAGram allows the use of linguistic priors to enhance morphological segmentation in a minimally supervised manner. Below we introduce two methods of including linguistic priors, specifically *grammar definition* and *linguist-provided affixes.*

3.4.1.4 Grammar Definition

A language-specific grammar is tailored for the language of interest by modeling specific morphological phenomena. While the grammars described in Table 2 are intended to be generic and to describe word structure in any language, we hypothesize that imposing LS constraints will be more efficient. Therefore, we investigate the incorporation of linguistic priors in the form of grammar definition, where we model LS morphological phenomena as part of the grammar. For example, we utilized the best on average performing grammar *PrStSu+SM* with Japanese as a case study using the following specifications: 1) A word has a maximum of one one-character or two-character prefix; 2) A stem is recursively defined as a sequence of morphs in order to allow for stem compounding; 3) Characters are separated into two groups, Kana (Japanese syllabaries) and Kanji (adapted Chinese characters); *SubMorph* represents a sequence of characters that is either in Kana or Kanji.

3.4.1.5 Linguist-Provided Affixes

In this approach, an expert in the underlying language compiles a list of carefully selected affixes and seeds it into the grammars prior to training the morphological-segmentation model using the *Scholar-Seeded* learning setting. Because they are high quality, the affixes are seeded as adapted non-terminals to encourage the instantiation of the corresponding subtrees. Here we use Georgian and Arabic as two case studies for the use of linguist-provided affixes. In the case of Georgian, a linguist who is an expert in Georgian as a second language, compiles a list of 119 affixes[14] that are collected from the leading reference grammar book [59]. For Arabic, a computational linguist who is a native speaker of Arabic compiles a list of 33 affixes.[15]

3.4.2 Unsupervised POS Tagging via Cross-lingual Projection

Unsupervised cross-lingual POS tagging via annotation projection uses available POS tags from a source language to project onto a target language using word-level alignments. The projected tags then form the basis for training a POS model for the target language. The overall pipeline is illustrated in Figure 8 with the left hand illustration outlining the word-based alignment process and the right hand illustration showing the stem-based alignment process.

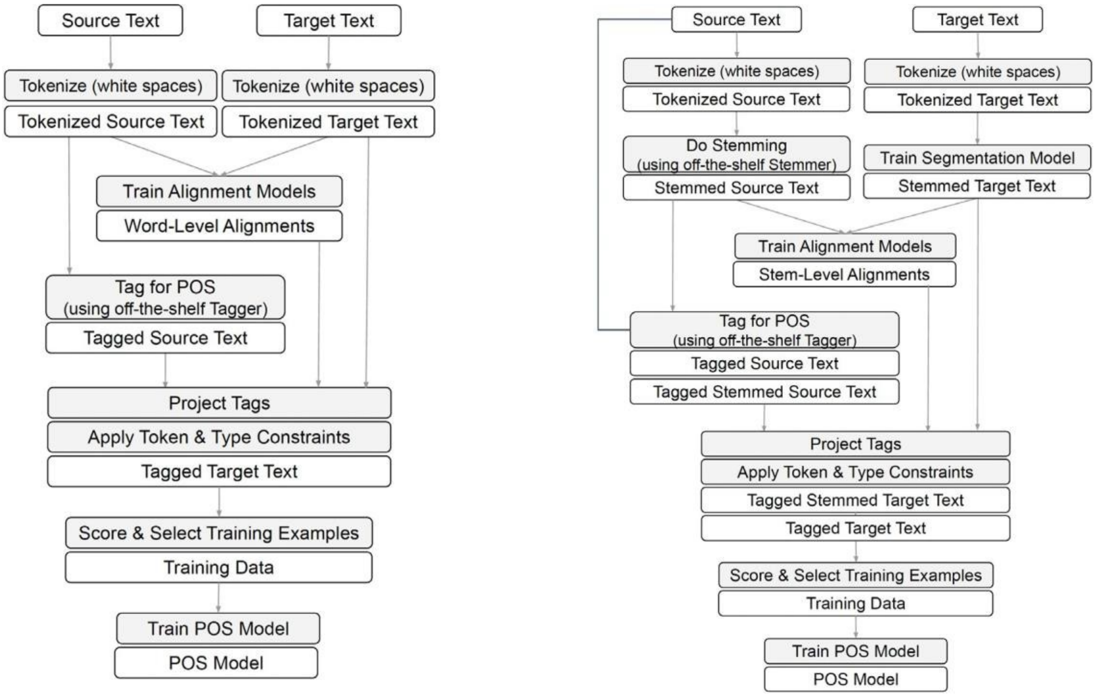


Figure 8. The Overall Pipeline for Unsupervised Cross-lingual POS Tagging via Alignment and Projection: Word-level Alignment (left); Stem-level Alignment (right).

3.4.2.1 Cross-lingual Projection via Word/Stem Alignments

We have developed a robust approach for standardizing the process of annotation projection by exploiting and expanding upon current best practices, in order to produce reliable annotations that improve the quality of the training data for the target language POS tagger. Our approach

[14] https://github.com/rnd2110/MorphAGram/blob/master/data/georgian/data/elk.txt

[15] https://github.com/rnd2110/MorphAGram/blob/master/data/arabic/data/elk.txt

includes: 1) the use of bidirectional alignments; 2) coupling token and type constraints on the target side; 3) scoring the annotated sentences for the selection of reliable training instances and 4) use both word-based and stem-based alignments. Below we describe the steps for cross- lingual projection used to build the training data for the POS tagging model.

3.4.2.1.1 White Space Tokenization

Starting with a sentence-aligned parallel text, we first perform white space tokenization on both the source and the target sets by separating punctuation marks and symbols into standalone tokens. We use *Stanza*[16] [60] to tokenize five of our six experimental source languages – specifically English, Spanish, French, German and Russian. For enhanced performance on Arabic white space tokenization, we use *MADAMIRA*[17] [61]. In order to keep our approach fully unsupervised, we tokenize the target language by applying a large set of language independent (LI) regular expressions using built in features of the Python programming language that can help recognize punctuation marks and symbols.

3.4.2.1.2 Word-Level Alignment

During OP1, we use the sentence-aligned parallel data to train bidirectional word-alignment models by aligning both the source and target texts at the word level in both directions. We experiment with two language-independent unsupervised word-level alignment systems, namely *GIZA++*[18] [62] and *Fast_Align*[19] [63]. As *GIZA++* consistently yields better results, we use it to align all of our target-source language pairs. Though our aim is to generate high-quality, one-to-one word-level alignments to use as the basis for annotation projection, word-level alignments suffer from non-precise translations, and there is no one-to- one correspondence between the words across parallel texts—this results in null, one-to-many and many-to-one alignments. To address this, we eliminate sentences of more than 80 tokens and only consider those instances where bidirectional word alignments (that is, one-to-one word alignments in both the source-to-target language and target-to-source language) exist. We also exclude alignments where the average of the bidirectional alignment probabilities is below a given threshold, α.

3.4.2.1.3 Stem-level Alignment

Word structures with many affixations increase the ratio of word types to word tokens, resulting in sparse alignment models and incomplete projections that form null assignments on the target side. These null assignments result in learning gaps for the POS model or scores too low for the underlying sentences to be used as training instances, reducing the overall quality of the POS model. As an example, we take Biblical verse *Matthew 15:35*, *"He commanded the multitude to sit down on the ground,"* and generate the word-alignment models for Arabic and Amharic trained on the New Testament. As shown in Figure 9, two Arabic- Amharic pairs are not aligned, resulting in null assignments. Using the stem instead of the word as the core unit of abstraction is more productive in such instances, as the stem is usually shared by all the members of a paradigm, minimizing misalignment. Figure 9b shows that stemming the Arabic and Amharic texts results in complete one-to-one alignments and projections. Assuming that the source language is high resource and has an off-the-shelf stemmer available, we can use

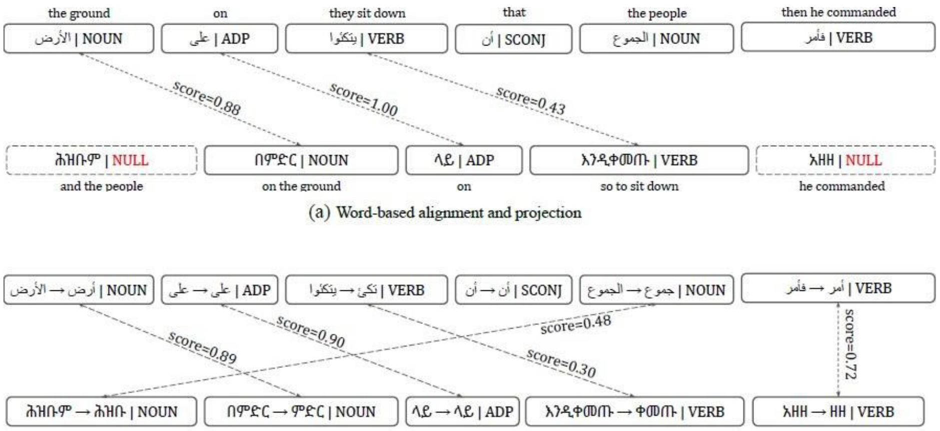
[16] https://github.com/stanfordnlp/Stanza

[17] https://camel.abudhabi.nyu.edu/madamira

[18] http://www.statmt.org/moses/giza/{\protect\protect\protect\edefTU{TU}\let\enc@update\relax\edefTimesNewRoman(0){TimesNewRoman(0)}\edefm{m}\ edefn{n}\protect\xdef\TU/TimesNewRoman(0)/ m/n/7{\TU/TimesNewRoman(0)/m/n/ 10}\TU/TimesNewRoman(0)/m/n/7\size@update\ enc@update\itshapeGIZA++}.html

[19] https://github.com/clab/fast_align

the Snowball Stemmer [64] as part of NLTK[20] [65]—as we do MADAMIRA. At the same time, we run MorphAGram (see Section 3.4.1, MorphAGram: Unsupervised Morphological Segmentation) in the Cascaded setting using two rounds of learning - first training a segmentation model using the language- independent high precision grammar *PrStSu2b+Co+SM* to obtain a list of affixes, and then seeding these affixes into the best performing language independent grammar, *PrStSu+SM*,for the second round. As discussed in Section 3.4.1, MorphAGram: Unsupervised Morphological Segmentation, both *PrStSu2b+Co+ SM* and *PrStSu+SM* model the word as a sequence of prefixes, a stem and suffixes, where the prefixes and suffixes are recursively defined in order to model multiple consecutive items. This process is suitable for morphologically complex languages.



(a) Word-based alignment and projection

(b) Stem-based alignment and projection

Figure 9. An Example of Alignment and Projection from Arabic to Amharic.
The alignment models are trained on the New Testament. Arabic reads right to left.

3.4.2.1.4 Source Language POS Tagging

Since cross-lingual projection requires a common POS tagset for all languages, we use the universal POS tagset of the Universal-Dependencies (UD) project,[21] which consists of 17 universal POS tags. After converting the output of the Penn Treebank (PTB)[22] tags into their universal cognates, we once again use *Stanza* to tag the source-side text— except in the case of Arabic, for which we apply MADAMIRA. However, since MADAMIRA was not designed to follow the UD guidelines, we correct the mapped Arabic analyses of the most frequent 2,500 POS-lemma pairs by manually selecting the most likely analysis for each pair.

3.4.2.1.5 POS Projection using Token and Type Constraints

Based on the mapping generated by the word-level alignments, we use token and type constraints first introduced by [66] to project the POS tags from the source onto the target language. Type constraints operate by defining the set of POS tags a word type can receive, which in a semi-supervised learning setup can be obtained from an annotated corpus [67] or from another resource that can serve as a POS lookup such as Wiktionary [68], [66]. To extract type constraints in an unsupervised fashion, we follow the approach proposed by [69], in which we accumulate the

[20] https://www.nltk.org
[21] https://universaldependencies.org/u/pos
[22] https://catalog.ldc.upenn.edu/LDC99T42

counts of the different source-side token POS tags that align with the target-side tokens of that word type, in order to define a tag distribution for each word type on the target side. The POS tags whose probabilities are equal to or greater than some threshold, β, then become the type constraints for the underlying word type. For token constraints, every target-side token gets assigned the POS tag of its corresponding source-side token.

In combining token and type constraints, we take a slightly different approach than [66] and [69]. Specifically, if a token is not aligned or its token constraint does not exist in the underlying type constraints, we give the token a NULL tag and it becomes unconstrained. Otherwise, the token constraint is applied and is used to represent the projected tags. In contrast to the previous work, we also do not use type constraints to impose restrictions while training the POS model, since this would restrict the performance of our neural architecture. Instead, we apply token and type constraints on the labeled stems on the target side when using stem-level alignments. Since we train the ultimate POS model on the word level, however, we ultimately replace each target stem with its corresponding word and assign the word the stem-based POS tag.

3.4.2.1.6 Selection of Training Instances

In supervised learning, adding more training instances generally improves the performance of the system until saturation is reached. For unsupervised learning, adding more training instances can introduce noise that actually limits the quality of the new model. As a result, we choose to score the target sentences based on their "annotation" quality *prior* to training a POS tagger using the projected tags as labels, and exclude the ones whose scores are below a threshold, γ. We define a sentence score as the harmonic mean of its density S_d and alignment confidence S_a, where S_d is the percentage of tokens with projected tags, and S_a is the average alignment probability of those tokens.

$$Score(S) = \frac{2(S_{\mathrm{d}} \cdot S_{\mathrm{a}})}{(S_{\mathrm{d}} + S_{\mathrm{a}})}$$

Filtering out sentences with low alignment confidence is crucial for training a high quality model, as demonstrated by previous research [70]. However, we add a density factor to this process in order to maximize the benefit our neural architecture derives from longer contiguous labeled sequences.

3.4.2.2 POS Tagging Models

We developed two POS tagging models: 1) an open-source, SOTA neural POS tagging model; 2) a POS tagger based on Averaged Perceptron, which is part of the SCRIPTS pipeline.

Neural POS Tagging. The architecture of our POS tagger is a bidirectional long short-term memory (BiLSTM) neural network [71]. BiLSTMs have been widely used for POS tagging, [72], [73], [74], [75], [76] and other sequence-labeling tasks, such as named entity recognition [77]. The input to our BiLSTM model is a sentence that has been automatically labeled through alignment and projection. The word representation, meanwhile, is the concatenation of four types of embeddings:

1. Pre-trained contextualized word embeddings
2. Randomly initialized word embeddings
3. Affix embeddings of 1, 2, 3, and 4 characters

4. Word-cluster embeddings

For the pre-trained contextualized word embeddings (1), we use the final layer of the multilingual cross-lingual LM (*XLM)-RoBERTa (XLM-R)* LM [78], relying on the average of the embedding vectors of the first and last sub-tokens of each word to represent its pre-trained contextualized embeddings, which yields better performance than using only the embeddings of the first or longest one token. To apply our neural architecture to a test language not represented in the *XLM-R* model, it is possible to train a custom *XLM* transformer-based LM[23] if monolingual texts and suitable computational resources (computing power and training time) are available- thus our architecture can be applied to languages beyond those present in the *XLM-R* model. The randomly initialized embeddings (2) rely on the target side of the parallel data and are learned as part of the training phase. Coupling randomly initialized and pre-trained embeddings is essential when the training data and the pre-trained embeddings are from different domains; in our learning setup, for example, we use the Bible text for training, while the *XLM-R* model is trained on texts from Wikipedia[24] and the CommonCrawl corpus. For affix embeddings (3), we use randomly-initialized prefix and suffix n-gram character embeddings, where n is in $1, 2, 3, 4$. Our experiments show that affix embeddings are more efficient for POS tagging than character embeddings across an entire word. For word cluster embeddings (4), we use hierarchical Brown clustering [79], producing clusters for each target language using Percy Liang's implementation[25] [80] on a monolingual text that combines the Wikipedia and Bible texts of the target language. Here, the Biblical text is white-space tokenized, while the fairseq library[26] is used to handle the scraping and cleaning of the Wikipedia data. For each word, we then concatenate the main cluster (that is, the binary representation of the corresponding leaf node) with all of its ancestors (the prefixes of the binary representation) in order to generate the embedding vector that represents the clustering information for the underlying word. This allows us to use the hierarchical clustering information without committing to a specific granularity level—a particular advantage where high level clusters may be insufficient and lower ones may reflect over clustering.

We compute the output using softmax activation on top of the BiLSTM encoding layer. Since some words receive null assignments, however, we set the value of the output neuron corresponding to the null tag to so that it does not contribute to the calculation of the softmax probabilities; this prohibits the model from decoding null values. Moreover, we mask those words with null assignments when computing the network loss.

3.4.2.2.1 POS Tagging Using Averaged Perceptron

An averaged perceptron is a basic version of a neural network, in which inputs are classified into several possible outputs based on a linear function. The outputs are then combined along with a set of weights derived from the feature vector that constitutes the "perceptron." Because good weight values will not change often, however, tracking the average values of the weights instead of the actual weight values after each learning pass is a preferred strategy. Averaged-perceptron models are suited to learning linearly separable patterns and have proved successful for fast and efficient sequence-tagging tasks, such as POS tagging. We use the averaged-perceptron implementation by Mohammad Rasooli.[27] We adapt beam search [33], [34] for the calculation of

[23] https://github.com/facebookresearch/XLM
[24] https://wikipedia.org
[25] https://github.com/percyliang/brown-cluster
[26] https://github.com/pytorch/fairseq
[27] https://github.com/rasoolims/SemiSupervisedPosTagger

the best POS-tagging sequence, in which we produce a ranked list of the top POS tags for each word in the sequence.

3.4.2.3 Multi-Source Projection and Decoding

The availability of parallel corpora that involve multiple languages encourages the use of multiple source languages for cross-lingual POS tagging via alignment and projection [81], [82], [83]. For example, the Bible [84] has complete translations in 484 languages and partial translation in 2,551 languages, which meets our low resource assumptions, as it is small in size and out-of-domain with respect to the evaluation sets. Here we introduce two multi-source approaches: 1) multi-source projection, where we combine tags projected from multiple source languages onto the target side prior to training the POS model; and 2) multi-source decoding, where we combine the tags produced by multiple single-source models to tag a given text in the target language.

3.4.2.3.1 Multi-source Projection

In this approach, we draw on the work of [81] and [82] to generate a projection from multiple source languages by using parallel corpora involving the target language and at least two others to derive multiple POS assignments on the target side. We begin by independently conducting alignment and projection—along with coupling token and type constraints - between each source language and the target one. This results in multiple POS assignments for the target text, where a token might receive one or more POS tags, including *NULL*. Then we apply a voting mechanism for each token in order to select the most likely correct POS tag out of those that have been assigned. The resulting annotations are then used to train a POS model, where the high scoring sentences are selected as training instances. We developed two voting mechanisms for the selection of the final POS assignment: 1) maximum voting and 2) Bayesian inference. In maximum voting (1), we take each target token and assign the projected tag that receives the maximum voting across the source languages, weighted by the alignment probability that corresponds to the underlying {token, source language} pair. We denote this voting setup by MP_{wmv}. In the Bayesian inference setup (2), we construct confusion matrices to identify which sources to rely on for specific sets of tags (MP_{bys}). We also conduct weighted Bayesian inference, where we combine both mechanisms in hybrid setups (MP_{wbys}).

3.4.2.3.2 Multi-source Decoding

Since we develop multiple POS models that correspond to different source languages, we can create a classifier ensemble that votes among the outputs of the different models on the token level. The main difference between this approach and multi-source projection is that, unlike the voting approach used in multi-source projection, the voting takes place as part of the decoding process *after applying the models* on some given text in the target language. Here we once again develop 1) maximum voting; and 2) Bayesian inference voting for the selection of the final POS assignment. For the former, we combine the tagging outputs of multiple POS models through weighted maximum voting that is similar to the MP_{wmv} setup.

Here, the weight is defined using two different techniques: alignment-based similarity (MD_{wmv_a}) and decoding probability (MD_{wmv_d}). Bayesian inference is similarly applied as for multilingual projection case, but instead of measuring the reliability of each source language for the assignment of each POS tag before training the POS model, we measure the reliability of each single-source model for the assignment of each POS tag after decoding the underlying text in the target language. We denote this setup by MD_{bys}. We also conduct weighted Bayesian inference, where we combine both mechanisms in hybrid setups ($MD_{wbys}a|d$).

3.4.3 SCRIPTS Normalization

SCRIPTS Normalization[28] is a normalization system that does text cleanup, transliteration and a number of other operations to control numbers, punctuation marks, repetitions and foreign text.

The system currently supports the following set of languages: English, Swahili, Tagalog, Somali, Lithuanian, Bulgarian, Pashto, Farsi, Kazakh and Georgian. The details of the system can be found in Appendix B.

3.5 CLIR

CLIR is accomplished by combining a set of sequentially run components that carry out a number of steps in order to support search across foreign language documents of various formats. First, indexing is performed on the following types of source documents: text documents in their original ("foreign") language, ASR transcripts of spoken foreign content, and alternative foreign words and phrases recognized by KWS (**indexing**). English indexes for both stems and words are also created using translation results for 1-best MT of text and speech, and translation probabilities are also stored for use with *probabilistic structured queries* and the *NNLTM* (**translation**). Both queries and documents are further enhanced using some combination of stemming, stopword removal, and query or document expansion, and those results are also indexed (**enhancement**). Query processing is also performed as a pre-processing step (**query processing**). The results of these steps are then used as input to several ranking methods to rank order documents (**ranking**). Different choices across these components result in diverse ranked lists of documents; these can then be combined into a single ranked list (**late fusion**).

During the BP (but not during OP1 or OP2), domain and language filtering were then applied (**filtering**). Finally, tuned cutoffs are applied to identify the most highly ranked documents, which are returned as the result set (**cutoffs**). Each of these components are discussed below.

3.5.1 Translation

We applied six techniques for crossing the language barrier. Query translation uses one-best MT to substitute a foreign language query for the original English query. Document translation does the reverse, substituting an English translation for a foreign language document. We also tried three complementary techniques for generating multiple alternative English translations of document terms: probabilistic structured queries, contextual lexical translation, or *n*-best MT. Our sixth technique used multilingual embeddings to represent queries and documents in a common feature space.

3.5.1.1 Query MT

Early experiments with dictionary-based query translation in the BP yielded low quality results, leading us to focus future query translation efforts on MT systems. In general, MATERIAL queries were not well-formed linguistic expressions, limiting the utility of the MT LMs we had available. Although we retained the capability to generate MTs of queries in the final system, we rarely made use of that approach in the submitted system combination results.

3.5.1.2 Document MT

By contrast, using MT approaches for document translation was quite successful. Both the

[28] https://github.com/rnd2110/SCRIPTS_Normalization

MATERIAL queries and the documents were available in English, a language for which very strong LM are available. We ultimately generated three translations for every document: one using SMT and two using NMT. In OP2, we generated specialized translation systems for spoken content that had been trained without punctuation in order to better match the output of the ASR system, which also lacked punctuation. Rather than combining the results of the three translation systems into a single translation, however, we used each one with a separate retrieval system, relying on late fusion to generate multiple translation benefits. One particular advantage of MT output over the other approaches we investigated is that word order is estimated; there- fore, we always used MT when our query processing involved phrase matching.

3.5.1.3 *N*-best Document MT

MT systems generate their best guess of what word should be inserted at each point in a translation. At the same time, all modern MT systems - whether statistical or neural - maintain internal representations of uncertainty that can be used to generate alternative translations. These alternatives can be generated either as alternative term sequences or as alternative terms at each point in the 1-best term sequence. Gaining benefit from alternative term sequences in an IR system is difficult, however, because the alternative sequences often have quite a small impact on meaning (e.g., substituting *a* for *an* in a 20-word sentence), while IR is concerned with possible substantive differences (e.g., substituting insect for travel when translating the word for fly). A better approach, then, is to generate alternative terms *at each point* in the sequence. We initially implemented this as a beam search in our SMT system and in one of the NMT systems; the benefits proved so substantial that we subsequently implemented it in our second NMT system as well. To use these multiple translation alternatives, we simply estimate the expected frequency of each English term in a foreign language document as the sum over the document language term instances of the probability that document language term would translate to the English term of interest, as shown in Equation 5.

$$\mathrm{tf}(t,d) = \sum_{k \in T(t)} \mathrm{tf}(k,d) \cdot p(t|k) \tag{5}$$

where *tf*(*t, d*) is the expected value of the frequency of English term *t* in document *d*, *tf* (*k, d*) is the actual frequency of foreign term *k* in document *d*, *T* (*t*) is the set of known foreign terms that can be translated as *t*, and *p*(*t k*) is the probability that foreign term *k* would be translated to English as *t*. Fortunately, this computation is fairly efficient because we can limit the beam width and use an index structure to focus the computation on cases that have a non-zero probability of translating to a query term.

3.5.1.4 PSQ

NMT systems rely on LM in both the source and target languages, whereas SMT systems require an LM for only the target language. While a strong LM generally improves the accuracy of the translation results, information retrieval's focus on selection means that rare terms have an outsized influence on IR results.[29] Of course, because rare terms appear infrequently in the training data, they tend to be the least well modeled by LM; as a result, we also wanted translation

[29] For example, an emphasis on rare terms have been hand-engineered into the BM25 and Query Likelihood ranking models that are described in Section 3.5.5, Ranking.

models available that didn't rely on LM. An SMT system is well suited to this task, especially since we can leverage multilingual dictionaries (e.g., Panlex) to mitigate sparsity effects for at least some of the words that are rare in running text. The resulting computation is identical to the way in which n-best MT is used - that is, using the PSQ [85] upon which n-best MT is actually based.

3.5.1.5 Contextual Lexical Translation

Document context is an important feature leveraged by NMT systems when creating translations of document terms. In a typical sequence-to-sequence (S2S) model, for example, target words are generated in a sequential manner using the source document context *and* the previous target words. (NNLTMs) [2], however, produce word translations conditioned only on the source word context, which is then used to perform CLIR. The NNLTM is trained using aligned sentences from parallel text to generate the translation probability for each word in a way that models the context of that word on the source (foreign) but not the target (English) side. Specifically, for an aligned word pair, f_i ↔ e_j, it uses a contextual window of k terms around the source word f_i to predict the target word e_j. NNLTM consists of an embedding layer that maps the $2k + 1$ source words to separate embeddings which are then concatenated and fed to a single feed-forward layer. The final layer then produces a softmax distribution of *contextual* probabilities $P\ (ej\ \ f_{i-k}..f_{i}..f_{i+k})$ over the target vocabulary and the model is optimized using cross-entropy loss using one-hot representation for the target word e_j.

3.5.1.6 Early Fusion

Generating translation probabilities from SMT and NMT systems both with and without source and/or target language context produces a diverse list of translation alternatives which can be combined to produce an aggregated list of alternatives that can be fed into a CLIR system. We refer to this form of evidence combination, which involves combining the outputs of different systems *before* retrieval, as *early fusion*. To conduct early fusion, we make use of *CombMNZ* [86], a widely used data fusion method which utilizes the scores of the documents returned by two or more systems. As detailed in Section 3.5.6, Late Fusion, CombMNZ uses the sum of document scores produced by different retrieval systems and multiplies it by a parameter n_d which denotes the number of systems that mark the document d as relevant to the query:

$$sc^{CombMNZ}(d) = n_d * \sum_{i=1}^{n} sc^{s_i}(d) \tag{6}$$

As a result, CombMNZ promotes the documents which are returned by multiple systems and can be expected to be especially helpful in our setup. Unlike post-retrieval combination, in-retrieval system combination approach directly combines word translation probabilities at query time.

Implementing this is straightforward in our setup, because we have two translation approaches, each of which provides n-best translations with assigned probabilities. For this, we use a variant of CombMNZ for the combination of context-dependent and context-independent translation probabilities:

$$sc^{CombMNZ}(w) = n_w * \sum_{i=1}^{n} sc^{s_i}(w) \tag{7}$$

In the preceding equation, $sc^{s}i$ (w) is the probability of the translation of the word w by the i^{th}

system, and n_w is a count of translation evidence sources that include the word. This helps in augmenting the computed weight for a translation that occurs in multiple sources. The combined weights are then used in the retrieval model explained below.

3.5.1.7 Multilingual Embeddings

Like all embeddings, cross-lingual or multilingual embeddings provide a way to cross the language barrier through the use of shared vector space in which the query and the document terms are mapped together such that terms with similar meanings have a similar representation. Below we describe the methods for creating embeddings that we experimented with in our ranking models:

- **fastText.** fastText [87] models learn monolingual representations for character n-grams from large, unlabeled corpora. These monolingual embeddings are converted to cross-lingual embeddings using an alignment approach known as Procrustes [88]. The idea is to align the vector spaces of the two languages closely using a bilingual dictionary consisting of pairs of lexicons. The entries in the lexicons are the anchor points; the Procustes approach involves bringing the representations of terms in the dictionary close together in the shared vector space. The cross-lingual embeddings for each term, however, does not depend on its context, so only one represent-tation is generated.
- **Multilingual Bidirectional Encoder Representations (mBERT).** mBERT is a multilingual extension of the BERT model [89] that generates a contextualized representation of terms learned from a large amounts of text in multiple languages. mBERT uses a shared encoder consisting of several layers of multi-head self-attention, commonly known as a transformer. mBERT is trained in a self-supervised manner using two pre-training tasks: 1) *masked language model* (MLM) which involves randomly masking tokens in a sentence and using the unmasked tokens to predict the masked tokens, and 2) *next sentence prediction* (NSP), which involves predicting whether a given sentence follows the previous sentence or not. These two tasks help in generating a representation of terms that is context-dependent.
- **XLM-R.** XLM-R [78] is another BERT-style model that shares the same design principle as mBERT. Two main differences between XLM-R and mBERT are 1) increased amounts of training corpora, and 2) large vocabulary size. XLM-R is available in two variants, a base version consisting of 12 layers of transformers and large version consisting of 24 layers of transformers. Similar to mBERT, XLM-R embeddings are generated using MLM as the pretraining task.
- **Aligning Word Embedding Spaces of Multilingual Encoders (AWESOME)-align.** awesome-align proposed by Dou and Neubig [90] seeks to fine-tune mBERT and align the contextual embeddings, while also improving performance on the task of word-pair alignment in parallel sentences using these embeddings. We find that after performing this fine-tuning, the cosine similarity of synonyms is improved across language pairs.

3.5.2 Enhancement

Query and document representations can be enhanced in several ways before performing ranking.

Here we describe stopword removal, stemming, to techniques for query expansion and two techniques for document expansion.

3.5.2.1 Stopword Removal

In some cases, we removed stopwords prior to indexing for efficiency reasons. Because as expected we found little effect on retrieval effectiveness from different stopword lists, we ultimately selected the Indri stopword list.

3.5.2.2 Stemming

English stemming of MT results generated both small improvements in average ranking quality and resulted in greater diversity when results were obtained with and without stemming. We also ran MT systems that were trained to generate stems directly; because these were less sparse in MT training, they helped contribute to diversity, but may have suffered more from ambiguity in LM. We also experimented with stemming in the foreign language using segmentation from the text processing team but found the improvements it offered unreliable. We also trained a range of PSQ matrices, with stemming on both sides (foreign and English), one side (foreign or English) or neither side. Stemming on English alone proved to be the best option.

3.5.2.3 Blind Relevance Feedback for Query Expansion

We used blind relevance feedback to expand flattened versions of the original English query. For this process, we indexed *The New York Times (NYT)* from 1987-2007, which contains 1.9M documents and 1B terms in total [91]. In OP2, we augmented this with the CommonCrawl News collection for 2018-2019, which yielded 5.3M documents and 3B terms in total. We indexed these collections using Indri, and the default Indri normalization, tokenization, and stopwords removal settings. A configurable number of terms were then selected from a configure- able number of highly ranked documents and used to expand the queries. As expected, this approach worked well for conceptual queries.

3.5.2.4 Embedding-based Query Expansion

An alternate query expansion approach is to project them onto a vector space and find the neighboring terms that are the closest to the encoded query vector. We achieve this by using word-embedding models that generate a distributed representation for every term in the query into a low dimensional vector space. For a given MATERIAL query, we encode the individual query terms separately using the Word2Vec [92] embedding model and use the average of the query vectors to produce an aggregated vector. This aggregated vector is then used to find the top-*k* closest neighboring terms in terms of the cosine similarity. These top-*k* terms are used to expand the query, with the cosine similarity indicating the corresponding weight in the retrieval models for CLIR.

3.5.2.5 Doc2query

Doc2query [93] is a SOTA model added in OP2 that uses BERT as a generative model for document expansion. For each document in the collection, the BERT model first predicts a list of queries that might be used to search the document and then concatenates these queries with the document. These new concatenated documents are then indexed and we search inside of this newly built index. As the lengths of the documents in the collections are modified, we follow the setup used in the original Doc2Query paper and use the collection with BM25 ranking, which we tune to optimally work with the lengths of the newly created documents.

3.5.2.6 DeepCT

Like Doc2query, DeepCT [94] is a SOTA model added in OP2 that alters the document representation prior to BM25 ranking; in this case, however, the BERT model predicts the optimal frequency of each term in each document, and then alters the index to use that frequency. This is done by creating a new document in which each word is repeated the optimal number of times. Unlike Doc2query, DeepCT doesn't expand the document by any words which are not in the original document. These newly built documents are then indexed and used in a search.

3.5.3 Indexing

We construct indexes to enable efficient access to the term and document features used in the ranking component.

3.5.3.1 Indri

Starting in the BP, we used Indri information retrieval system [95] both to index documents in the foreign language and after MT. The Indri index supports ranking using both the full Indri query language and provides an Application Programming Interface (API) for use with other ranking systems. Character normalization was typically performed prior to indexing, and the index was configured to support both stemming we did using the Porter stemmer and the sequential dependence model we created through positional indexing.

3.5.3.2 Anserini

In OP2, we also indexed documents using Anserini [96], an iIR system built on Apache Lucene.[30] Using Anserini was convenient for specific purposes; notably for DeepCT and doc2query. For this, we relied on the default settings, which included stopword removal and stemming using the Porter stemmer.

3.5.3.3 Specialized Indexes

Starting in OP1, we designed and built additional indexes to support specific ranking models that are described in Section 3.5.5, Ranking - notably, PSQ-based HMM, Probability of Term Occurrence (PTO), and KWS. We built this indexing infrastructure from the ground up in Python and stored the pre-processed document collection statistics as serialized (e.g., pickle or Hierarchical Data Format 5 [HDF5]) Python objects.

3.5.4 Query Processing

Because all queries are known at the start of experimentation, we generate all needed versions of the queries once, as a preprocessing step.

3.5.4.1 Query Parsing

Query parsing was performed by the Query Analyzer docker component. The off-the-shelf Java library called ANother Tool for Language Recognition (ANTLR)[31] was used for query parsing and query validation utilizing the IARPA-provided Context Free Grammar (CFG). The results of query processing were formatted as a complex JavaScript Object Notation (JSON) object containing extensive information about a query, including: the list of query terms; which constraints are present; and whether it is a conjunctive, lexicalor conceptual query, among many other such

[30] https://lucene.apache.org/

[31] https://www.antlr.org/about.html

attributes. A separate JSON file was generated for each query and used by both the CLIR and Summarization (Section 3.7.1, Unsupervised Approach) components.

3.5.4.2 Flattening

An important component of the JSON object generated by the Query Analyzer was the different versions of the query that might be used by various system components. We referred to the process of creating these query versions as “flattening,” as these versions generally had fewer structural elements than the original query. These query variants included:

- A bag of words query that included content terms from the full query, including any specified synonyms, hypernyms, event frames or example_of part constraints. We describe this version as “flat.” As an example, the source query "bracelet[hyp:accessory], occupation[syn:job]" would yield the flat query "bracelet accessory occupation job."
- A bag of words query that included content words from the query *other* than those in synonym, hypernym, event frame or example_of constraints. We (somewhat arbitrarily) describe this version as “radically flat.” Here, the original query “bracelet [hyp:accessory], occupation [syn:job]" would yield the “radically flat” query "bracelet occupation."
- Other versions included a bag of words query that included content terms and synonym constraints, but not words from hypernym, event frame or example_of constraints. In this case, the query "bracelet [hyp:accessory], occupation[syn:job]" would flatten to "bracelet occupation job."

3.5.4.3 PSQ

The query JSON object also included versions of the query that contained translations and their associated translation probabilities for use with PSQ. The translation probability matrix used in the PSQ matcher is fixed at indexing time, so these queries could be pre-compiled for efficient execution at ranking time.

3.5.5 Ranking

Ranking is at the heart of CLIR; its goal is to assign a score to each document (or, in one case, to each pair of documents) for each query that can then be used to sort the documents in a best first order for that query.

3.5.5.1 BM25

BM25 is a pointwise lexical ranking function [97] that was originally designed for conceptual queries. Pointwise functions operate by first independently computing a real valued retrieval status for each document; documents are then scored in descending order of retrieval status value. BM25 retrieval status values are based on four factors that are computed independently for each query term:

- $\mathrm{tf}(t,d)$, the number of times each query term appears in a document, where increasing counts indicate increasing “aboutness”
- $\frac{l_d}{L}$, the number of terms that appear in a document, relative to the average number of terms in a document over the entire collection, where increasing ratios indicate *decreasing* “aboutness”
- $\mathrm{df}(t)$, the number of documents in which each query term appears, where increasing counts indicate decreasing specificity

The aggregate retrieval status value is then the sum of the term-wise values, as shown in Equation 8.

$$\mathrm{BM25}(q,d) = \sum_{t \in q \cap d} \log \frac{N - \mathrm{df}(t) + 0.5}{\mathrm{df}(t) + 0.5} \cdot \frac{\mathrm{tf}(t,d) \cdot (k_1 + 1)}{\mathrm{tf}(t,d) + k_1 \cdot \left(1 - b + b \cdot \frac{l_d}{L}\right)} \tag{8}$$

Applying BM25 effectively depends on the way the two aboutness measures are shaped and combined using two parameters, (k_1 and b). On average, BM25 offers substantial improvements over earlier approaches (often referred to generically as TF-IDF) for using similar document information as a basis for point- wise ranking. In the BP and OP1, we used a hand-coded BM25 implementation; in OP2 we also used the implementation in the Anserini IR system [96].

3.5.5.2 HMM

We use a two-state HMM [98] to estimate the relevance of document given an input query. The first state θ_e, generates English words, while the second state, θ_d, generates foreign words. Each English query q may consist of N terms $t_1, ..., t_N$.

The generation of query q can then be expressed as shown in Equation 9 where f is a foreign language word and ϵ enables basic document length normalization.

$$p(q|doc) = \prod_{n=1}^{N} \left[\alpha P(t_n^{(e)}|\theta_e) + (1-\alpha) \sum_{f \in t_n^{(f)}} P(t_n^{(e)}|f) P(f|\theta_d)^{\epsilon} \right] \tag{9}$$

The probability of generating foreign word f from state θ_d can be estimated from the counts shown in Equation 10.

$$P(f|\theta_d) = \frac{c(f, doc)}{\sum_f c(f, doc)} \tag{10}$$

The probability of generating English word $t_n^{(e)}$ from state θe is similarly estimated from counts in a large corpus of English (specifically, the Google one billion word collection [99]). We chose α as 0.1, thereby assigning higher weights to the second state (θ_d) as compared to the first state (θ_e). We set ϵ to 0.2 as this allows us to mitigate the effect of document length on the retrieval.

We only use words from the primary part of the query (that is, we exclude words in the synonym, hypernym, event frame and example_of fields) and we compute the relevance score as defined in Equation 9. The translation probabilities are obtained from aligning the parallel corpus in the respective MATERIAL language.

3.5.5.3 Query Likelihood Model (QLM)

An alternative approach to pointwise lexical ranking is to model both queries and documents as unigram LM and then to compute the probability that the query LM is generated from the document LM [100]. Dirichlet smoothing [101] is used to prevent numerical problems resulting from zero counts, which relies on the number of times a query term appears in the collection - rather than the number to documents in which a query terms appears - as the evidence for term specificity. As a result, while this approach relies on the same evidence as BM25, the QLM can yield somewhat different document rankings. For this work, we used the QLM implementation in the Indri IR system [95]

3.5.5.4 Sequential Dependence Model (SDM)

To model term order for phrase-based MATERIAL queries, we leverage the Indri SDM [102], a variant of QLM that additionally rewards terms appearing in a similar order in the query and the document. This evidence is aggregated with weights assigned to each contribution to produce the final ranked list of documents. In some cases, we also used a simpler version of phrase processing in which we required translations of query terms to be within two words of each other.

3.5.5.5 PTO

This model is identical to that defined by Equation 9, except that the foreign state model is switched by the probability of term occurrence [2], as shown in Equation 11.

$$p(q|doc) = \prod_{n=1}^{N} \left[\alpha P(t_n^{(e)}|\theta_e) + (1-\alpha) \left(1 - \prod_{f \in doc} (1 - p(t_n^{(e)}|f) \right) \right] \tag{11}$$

We once again exclude words except those from the primary part of the query and compute the relevance score for all the queries as defined in the Equation 11. The translation probabilities are obtained from aligning the parallel corpus in the respective MATERIAL language.

3.5.5.6 Searching *N*-Best Speech

In addition to 1-best hypotheses, ASR systems provide a set of alternative results that are compactly encoded in the form of lattices. Formally, lattices can be viewed as weighted finite state automata, which, though useful, are highly inefficient to search. To overcome this problem, the work in [103] introduced a procedure to convert automata into a weighted finite state transducer (WFST), which makes search faster. To generate the WFST, the lattices generated by an ASR system are first processed to create word indices for in-vocabulary (IV) search and phone-tic indices for out-of-vocabulary (OOV) search. The output label for each arc in the lattice is then supplied with the timing information. As a result, when the input query is supplied to the system, it returns a list of arc sequences with the time spans that match the query with an associated score, where the scores are the posterior probabilities of the arc sequences. In the technique we refer to as KWS, we use the posterior probability of each term to compute the expected counts [104].

For word *k* and document *d*, the expected count is shown in Equation 12, where *a* is a lattice arc, *l*(*a*) is a term label associated with a, *u* is a segment of document audio and *O*(*u*) are associated observations used by an ASR system to yield posterior probabilities.

$$\mathbb{E}(k|d) = \sum_{u \in d} \sum_{a:l(a)=k} P\left(a|O(u)\right) \tag{12}$$

3.5.5.7 Neural Reranking

In monolingual retrieval applications, the current SOTA approach involves building a retrieve- and-rerank pipeline. A typical pipeline consists of an initial candidate generation step where a fast retrieval system such as BM25 is used to produce the set of documents; these are then reranked by trained neural rerankers in one or more stages. The neural rerankers are initialized with contextualized LM that are pre- trained on large amounts of training data. Such pipelines have better performance than the traditional lexical/keyword-based retrieval models such as BM25 and QLM. We built a similar pipeline for CLIR, using a strong lexical baseline model (e.g., PSQ) for the initial stage and a neural CLIR reranker initialized with a multilingual pretrained model (e.g., XLM-R). By translating documents from MATERIAL languages to English, we are also able to leverage applying existing monolingual pipelines in English.

3.5.5.8 Zero-shot ColBERT-X

Dense retrieval models are a type of neural ranking model that encode the queries and documents separately into a shared vector space. This process allows the document representations to be precomputed offline and stored in specialized data stores supporting fast retrieval at query time, such as Approximate Nearest Neighbor (ANN) indexes. ColBERT [105] is a multi-stage dense retrieval model that computes multiple representations for the query and the document (one for each term) independently. The first stage involves generating candidate document sets by querying the ANN index built from document term representations using the query term representation. In the second stage, these candidate documents are reranked using a MaxSim operation, which computes the dot product between every query and document term to compute a final relevance score for the document.

3.5.5.9 Reranking with Position-Aware Convolutional Recurrent Relevance Matching (PACRR) and Pooled Similiarity (POSIT)

We used deep bilingual query document representations to boost cross-lingual document retrieval performance. We match queries and documents in both query and document languages with four components. By including query likelihood scores as extra features, our model effectively learns to rerank the retrieved documents using a small number of relevance labels for each of the LRL pairs. As shown in Figure 10, our model outperforms the competitive translation-based baselines on English-Swahili, English-Tagalog, and English-Somali CLIR tasks.

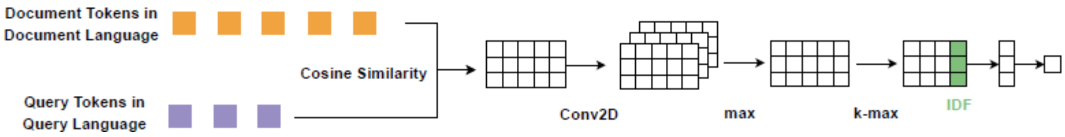


Figure 10. Bilingual PACRR-Deep Relevance Matching Model (DRMM).
Bilingual PACRR is the same except it uses a single Multilayer Perceptron (MLP) at the final stage.

In CLIR, given a user query in the query language *Q* and a document in the document language *D*, the system computes a relevance score *s* (*Q, D*). As shown in Figure 11, our model first translates the document as *D*ˆ or the query as *Q*ˆ, and then it uses the four separate components to match: (1) query language query with document language document, (2) query language query with query language document, (3) document language query with query language document, (4) document language query with document language document.

The final relevance score combines all components according to Equation 13, using un-weighted combination because we lacked sufficient training data to learn data specific component weights.

$$\bar{s}(Q, D) = s(Q, D) + s(Q, \hat{D}) + s(\hat{Q}, \hat{D}) + s(\hat{Q}, D) \quad (13)$$

To implement each component, we extended three SOTA *term interaction models*: PACRR as proposed in [106] along with POSIT-DRMM and PACRR-DRMM proposed in [107].

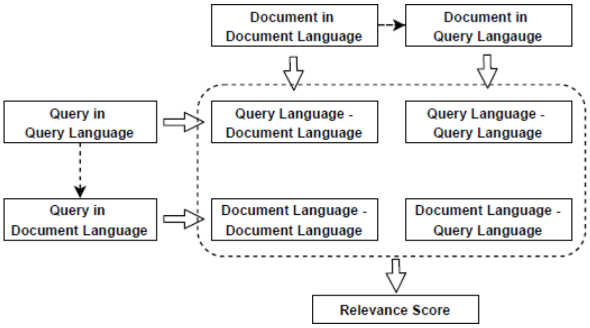

Document in Document Language
Document in Query Langauge
Query in Query Language
Query Language - Document Language
Query Language - Query Language
Query in Document Language
Document Language - Document Language
Document Language - Query Language
Relevance Score

Figure 11. Cross-lingual Relevance Ranking with Bilingual Query and Document Representation.

In term interaction models, each query term is scored to a document's terms from the inter- action encodings, and scores for different query terms are aggregated to produce the query-document relevance score.

The POSIT-DRMM model is illustrated in Figure 12. We first use bidirectional LSTMs [71] to produce context sensitive encodings of each query and document term. We then add residual connections to combine the pre-trained term embedding and the LSTM hidden states.

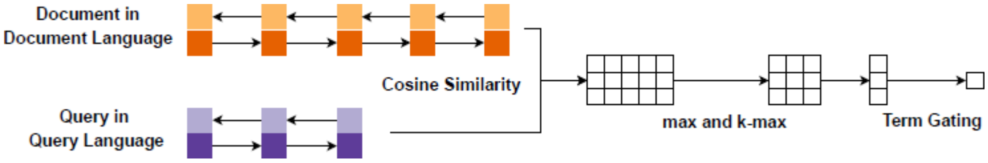


Figure 12. Bilingual POSIT-DRMM.

For the query language query and document term, we use the pre-trained word embedding in the query language. For the document language query and document term, we first align the pre-trained embedding in the document language to the query language and then use this cross-lingual word embedding as the input to the LSTM. Thereafter, we produce the document aware query term encoding by applying max pooling and *k*-max pooling over the cosine similarity matrix of query and document terms. Finally, we then use an MLP model to produce term scores. The relevance score is a weighted sum over all terms in the query with a term gating mechanism. More details about this approach can be found in [108].

The models that we used, Bilingual PACRR and Bilingual PACRR-DRMM, are illustrated in Figure 10. In implementing these, we first align the word embeddings in the target language to the query language and then build a query document similarity matrix that encodes the similarity between the query and document term. Depending on the query language and document language, we construct four matrices, $\mathrm{SIM}_{Q,D}$, SIM$Q,D^\wedge$, SIM$Q^\wedge, D^\wedge$, SIM$Q^\wedge,D$, one for each of the four components. Next, we apply CNNs over the similarity matrix to extract *n*-gram matching features, followed by max-pooling and *k*-max-pooling to produce the feature matrix where each row is a document-aware encoding of a query term. The final step computes the relevance score: Bilingual PACRR uses an MLP on the whole feature matrix to get the relevance score, while Bilingual PACRR-DRMM first uses an MLP on individual rows to get query term scores and then use a second layer to combine them.

3.5.5.10 Aligning Multilingual Contextual Embeddings

In a follow-up document reranking experiment during OP2, we used aligned multilingual contextualized embeddings, in which we used explicit cross-lingual alignment techniques to improve multilingual BERT representations. We compared the new, explicitly aligned rerankers with a baseline reranker on two MATERIAL datasets (Farsi and Kazakh) and on two datasets (Finnish and German) from the Conference and Labs of the Evaluation Forum (formerly known as Cross-Language Evaluation Forum [CLEF]). In all cases, we use English queries. The results showed that given a sufficient amount of parallel text, (1) these cross-lingual neural rerankers can outperform a PSQ baseline, and (2) that additionally leveraging cross-lingual alignments can lead to improved ad-hoc CLIR, compared to using baseline mBERT representations.

To do this, we tried two recently proposed fine-tuning-based alignment methods for mBERT embeddings. We focused on fine tuning embedding alignment procedures as 1) they are faster than off-line embedding alignment procedures like orthogonal transformations that must be ap- plied to each word representation 2) previous work has suggested that fine tuning based alignment is better suited for semantic tasks, and rotation-based alignment is more suited for structural tasks [109]. Since our task is an IR task, we opted for fine tuning alignment, which does not require any additional alignment computation after the completion of the fine-tuning step.

We worked with awesome-align [90] because their alignment objectives draw on LM techniques and include a sentential-level alignment objective. They use a combination of several objectives in order to perform fine tuning. The first is an MLM objective as shown in Equation 14.

$$L_{MLM} = \log p(x|x^{mask}) + \log p(y|y^{mask}). \quad (14)$$

Just as in BERT's pre-training step, in each of the parallel sentences, x and y, 15% of tokens are randomly masked by either a special [MASK] token, a random token, or are not replaced, and the model must fill the correct tokens back in. A translation language modeling (TLM) objective, as shown in Equation 15, is also used.

$$L_{TLM} = \log p(x; y|x^{mask}; y^{mask}) + \log p(y; x|y^{mask}; x^{mask}) \quad (15)$$

By allowing the model to perform MLM on parallel data, this further pre-training allows the model to better align its representations of the two languages. A parallel sentence identification (PSI) loss is also employed, in which the model must properly label a randomly selected pair of sentences from the corpus (x', y') as either true parallel sentences ($l = 1$) or not parallel sentences ($l = 0$), as shown in Equation 16.

$$L_{P\ SI} = l \log(s(x', y')) + (1 - l) \log(1 - s(x', y')) \quad (16)$$

The final fine-tuning objective used for pretraining mBERT for better aligned contextual embeddings is thus the combination of all these objectives summed over all sentences in the parallel corpus, as shown in Equation 17.

$$L = \sum_{x,y} L_{MLM} + L_{TLM} + L_{SO} + L_{PSI} + L_{CO} \quad (17)$$

While the first method from Cao et al. [110] is a relatively simple objective for alignment that can also be applied to static word embeddings, awesome-align employs several specialized LM objectives which are components of the pretraining process of mBERT.

We use each of these objectives to fine-tune mBERT for each language pair, thereby more closely aligning the contextualized representations of the two languages in each language pair. We then pass these models into the Contextualized Embeddings for Document Ranking (CEDR) architecture [111] so that these fine-tuned contextualized embeddings can be employed for document and query representations to determine relevance.

3.5.6 Late Fusion

Different ranking methods, used with different translations, different query versions, and/or different enhancement techniques, help generate diverse lists of ranked documents. Using late fusion (i.e., fusion performed after ranking), allows us to benefit from this diversity. For each query, the System Combination module receives a set of lists of ranked documents on the input, along with scores for each document, and produces a single ranked list of documents. We have experimented over the course of the program with a range of methods for this purpose, some of which use only the ranks of the returned documents, and others of which also used the score associated with each document. A detailed description of these methods (apart from Hierarchical Reciprocal Rank Fusion) can be found in [112].

3.5.6.1 Rank-Based Fusion

We used two approaches to rank-based fusion, one based on Borda counts [113] and the other based on reciprocal ranks. Borda-count fusion uses positional voting to assign a score to each document returned by the ranker as the number of returned documents - rank of the returned document. The scores for the document are then summed over all the rankers. In Reciprocal Rank Fusion (RRF) [114], the scores are calculated as 1 / rank of the returned document. The trec_tools [115] implementation of the reciprocal rank fusion uses an adjusted ratio of 1 / 60 + rank of the returned document which removes a disadvantage of the documents with very low ranks. Again, the scores are then summed over all the rankers.

3.5.6.2 Unweighted Score-Based Fusion

We used two approaches to unweighted score-based fusion, CombSUM and CombMNZ. The CombSUM method simply takes the scores of the document retrieved by all the rankers which are combined. CombMNZ is a refinement of this approach in which the summed score is then multiplied by the number of systems, ensuring that the documents returned by more systems are promoted. In either case, the scores from different systems can be in very different ranges and so need to be normalized and put into a shared range. For this, we have used sum-to-one (STO) normalization: the scores of all documents returned by particular systems were summed, and the score for a particular document is then divided by the sum.

3.5.6.3 Weighted CombMNZ

Weighted CombMNZ further modifies CombMNZ by assigning a weight to each system - the final score is then calculated as a weighted linear combination of the scores. We tuned the weights on development data, using the Maximum Query Weighted Value (MQWV) of each system on the DEV or DEV+ANALYSIS collection as its weight.

3.5.7 Filtering

In the BP, only documents in a specified domain and language were to be returned, which CLIR implemented by filtering the retrieved documents. This filtering was implemented as a part of the evidence combination component and was run after the combination of the documents but be- fore an application of the cutoff.

3.5.7.1 Domain Filtering

We build our own domain classifier to identify the domain of the document. This was a simple unigram Support Vector Machine (SVM) classifier built with the Weka ML toolkit, which used the Linguistic Data Consortium (LDC) *NYT* collection [91] for training. The categories of the articles were manually assigned domains (e.g., military, government and politics, law and order). All the articles in the selected categories were then used to train the classifier, which in turn was used to decide whether the document is in the domain; out-of-domain documents were simply not returned. As the *NYT* categories are relatively broad, the performance of the classifier was limited, which reduced the performance of CLIR with domain filtering.

3.5.7.2 Language Filtering

Documents for which the detected language did not correspond with the given language were filtered out. The language was recognized during the text pre-processing or during the speech recognition. Because speech recognition also provided a confidence level for each document's language membership, only documents with classified as having a non-matching language with a confidence level above a certain cutoff level were filtered out. Compared to the domain classifier, the language identification worked very well - even achieving 100% accuracy in some cases. This was therefore very helpful for CLIR.

3.5.8 Cutoffs

Up to this stage, we have been working on producing the best possible ranked list of documents. By the probability ranking principle [116], selecting an appropriate cutoff in an optimal ranking (i.e., one ranked in decreasing order of probability or relevance) would result in optimizing a cost function with the structure of Average Query Weighted Value (AQWV). Thus, we can approximate the optimal set by approximating the optimal ranking (as we have done here), and then approximating the optimal cutoff given that ranking.

3.5.8.1 Fixed Rank Threshold

Using a fixed rank strategy, an optimal cutoff value is estimated for the selected combination on the development set—typically, DEV+ANALYSIS. There are two major downsides of this approach: 1) the same number of documents will be returned for each query and 2) the observed density of relevant documents in the development set may differ from the actual (but unobserved) density of relevant documents in the evaluation set and because of the (observable) difference in the number of documents in the two collections. When an estimate of the relative density of relevant documents is available, however, the fixed rank cutoff can be scaled linearly by both factors.

3.5.8.2 STO Threshold

The STO cutoff strategy can select a different number of documents based on the complexity of each query. STO refers to STO normalization that is applied to the document scores. To start, the optimal ranked threshold which results in the optimal AQWV on the development collection (e.g., DEV+ANALYSIS) is first determined. The STO cutoff score is then estimated on the scores of the returned documents by aiming for this target ranked threshold. STO scores on the EVAL collection will naturally be scaled differently, but it is possible to select an STO score cutoff on the EVAL collection that results in the desired ranked threshold, and that is the approach that we use. As with the fixed rank cutoff approach, when the density of relevant documents is expected to differ systematically between the development and EVAL set, an estimate can be generated and used as a linear scaling factor; in this case, it is applied to the ranked threshold.

3.5.8.3 Query Specific Threshold (QST)

If the score for each document is its probability of relevance, then QST could be analytically derived from the AQWV definition [117]. We thus rescale the scores in a manner that provides a reasonable estimate of the probabilities, learning the mappings on the development collection and then applying them on the EVAL collection. When these estimates are accurate, QST generates very accurate cutoffs. Because these estimates are sometimes poor, using QST alone is unsafe.

Formally, this method tries to optimize the definition of the AQWV score shown in Equation 18, where *pMiss* is the average per-query miss rate, *β* is given in the MATERIAL program and set to 20, respectively 40 in the later stages of the project, and pFA is the average pre-query false alarm rate [117].

$$AQWV = 1 - pMiss - \beta * pFA \tag{18}$$

Then, it is then possible to calculate an optimal threshold using Equation 19, in which *C* is the size of the collection and N_{true}(*query*) is the number of the documents that are truly relevant to the query [117]. As this true number is really unknown, it needs to be estimated.

$$threshold(query) = \frac{\beta N_{true}(query)}{|C| + (\beta - 1)N_{true}(query)} \tag{19}$$

3.5.8.4 Average with Clipping

In the BP, we used a fixed rank threshold. We added STO and QST threshold in OP1 and generally averaged two or three of these estimates when selecting the rank cutoff for each query. Because QST can occasionally produce very high cutoffs and false alarm control is important when optimizing for AQWV, in OP2 we imposed a hard limit on the cutoff for any query. We set this, somewhat arbitrarily, at three times the fixed rank cutoff.

3.5.9 Formative Evaluation

The complexity and nuance of human language use makes IR largely an empirical discipline. Both traditional and neural ranking techniques benefit very substantially from training, with neural techniques benefiting from even a limited amount of task-specific fine tuning. In contrast to the summative evaluations conducted by the MATERIAL program, this dependence on tuning and training raises the question of how formative evaluation should best be conducted.

3.5.9.1 Test Collections

MATERIAL provided two types of test collections for use during system development, known as the ANALYSIS and DEV collections. In BP and OP1, the DEV and ANALYSIS collections were quite small, typically around 300 documents. Broadly speaking, the DEV collection was intended to be representative of the EVAL collection on which summative evaluation would later be conducted. The ANALYSIS collection, by contrast, was intended for exploring the effects of specific phenomena, and thus was not intended to be distributionally similar to the EVAL collection. In general, the value of a test collection increases with the number of positive relevance judgments in the test collection, because positive judgments are the minority class. During the BP and OP1, combining the DEV and ANALYSIS collections roughly doubled the number of positive judgments in the combined collection compared to the DEV collection alone, although at the cost of some loss of distributional similarity to the EVAL collection. Our experience in the BP indicated that using DEV+ANALYSIS as a basis for formative evaluation was generally superior to the use of DEV alone, and thus we used DEV+ANALYSIS routinely for formative evaluation in OP1. In OP2, the program reallocated evaluation resources to very substantially increase the size of the DEV collection; therefore, we used the DEV collection as a basis for formative evaluation in OP2. This change may also have improved comparability across research teams, as we believe that neither of the other two teams routinely been using DEV+ANALYSIS as a basis for formative evaluation.

3.5.9.2 Evaluation Measures

Because our approach relied on first ranking and then selecting a rank cutoff, we needed a suitable evaluation measure for optimizing ranking directly. We chose Mean Average Precision (MAP) for this purpose because the measure is strongly head-weighted (that is, influenced strongly by the top most rankings, which is where our selections will ultimately draw from) and because its design is well matched to the binary relevance judgments. MAP is the mean across the queries of the average across the relevant documents of the precision at the position in the ranked list of each relevant document. In that definition, precision is the fraction of the documents at or above the rank of a relevant document that are relevant. Relevant documents that are not found are modeled as being at infinite rank (i.e., as contributing 0 to the average of the precision values). As expected, we found that ranking systems with higher MAP also yielded higher AQWV given a suitable rank cutoff. MAP also has the desirable characteristics of being well normalized (it is bounded between

0 and 1) and of degrading gracefully as relevance judgments are ablated. Notably, if only a single relevant document were to exist for each query, MAP would be equivalent to the Mean of the Reciprocal of the Rank at which that one relevant document is found.

3.5.9.3 Zero-Relevant Queries

The EVAL collections used for summative evaluation were initially designed to intentionally avoid queries for which there were no relevant documents, although the separation of summative evaluation into separate text and speech evaluations after the BP did result in some queries with no relevant documents in one collection or the other. Queries with no relevant documents were, however, quite common in the DEV and ANALYSIS collections, and even the DEV+ ANALYSIS collection had a considerable proportion of zero relevant queries. By convention, MAP treats the contribution of a query with no relevant documents as zero, so these zero relevant queries simply reduce the measured MAP for every system. In order to preserve the full dynamic range for MAP (i.e., where a MAP value of one indicates perfect retrieval), we removed any queries for which there were no relevant documents when computing MAP during formative evaluation. That decision did not affect the preference order between systems, but it did have the potential to affect ratio comparisons. In retrospect, this proved a problematic choice because we initially reported formative evaluation results over a different query set than was being reported by the other teams in the one- to two-week evaluation "sprints."

3.6 Real-Time Demo

The system we have described was designed for flexible experimentation, and it proved very well suited for that purpose. However, when we first built a real time demonstration system, the query processing time was far too slow as systems optimized for speed need a different architecture from those designed for flexible experimentation. We thus harvested some of our component designs and reimplemented them with low query latency as the goal. For this, the major components we used are as follows:

3.6.1 Query Analysis

To improve response time, the Query Analysis component generated only two representations. For MT search, we used flat queries (a bag of words query including all query terms, including terms from synonyms, hypernyms, event frames and example_of constraints). For PSQ, we used radically flat queries (queries with all query terms, but *not* those in synonym, hypernym, event frame or example_of constraints).

3.6.2 Query Expansion

For conceptual and example_of queries, we used the embedding-based query expansion technique described in Section 3.5.2, Enhancement, to Expand the English queries.

3.6.3 Matching

To balance latency and effectiveness, we opted for two diverse systems: 1) PSQ, implemented with the Indri LM, and 2) BM25 implemented with ElasticSearch using the 1-best MT output from the Edinburgh MT system.

3.6.4 Late Fusion

The late fusion technique we used was the CombMNZ technique described in Section 3.5.6, Late Fusion.

3.7 Summarization

The role of summarization in this work is to create a short, compelling summary of a document selected by the upstream systems (i.e., CLIR) and assist an end user in determining quickly whether it is relevant to their given query. Ideally, the document is relevant, and the summary's content will focus on the query; however, the summarizer must also faithfully represent irrelevant documents as irrelevant. We use extractive summarization, which generates a summary by selecting sentences verbatim from the input document (as opposed to generating new sentences).

3.7.1 Unsupervised Approach

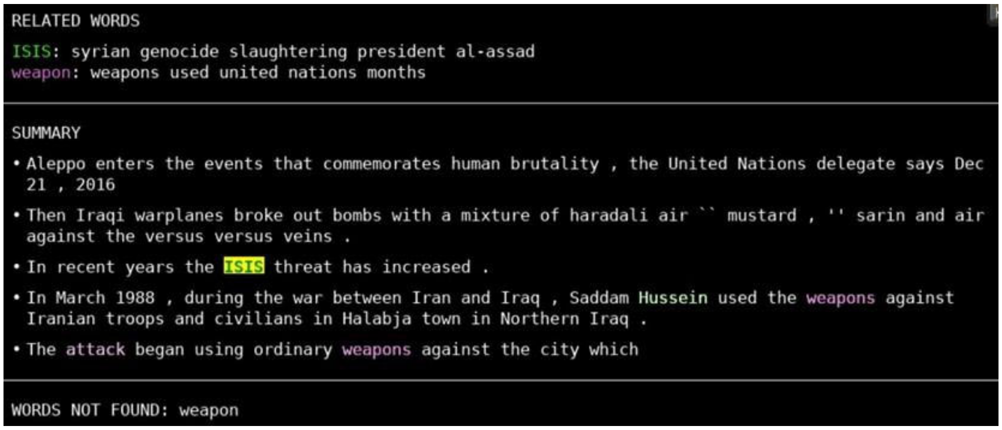
Our summarization work in the BP relied on unsupervised methods because we did not have any training data. For each relevant document returned by CLIR, we ranked all sentences by their similarity to the query using three methods. First, we ranked them using cosine similarity be-tween the English query embedding and an embedding for each of the document sentences.

The second ranking relied on the cosine similarity of the English query after it had been translated into the source language, against an embedding of each document sentence in the source language. Finally, we expanded the English query using query expansion as described in Section 3.5.4, Query Processing, resulting in a bag of words and did a direct match against the bag of words corresponding to each document sentence. The ranks resulting from these three matches were then merged using the Borda count algorithm [118] and sentences were sorted into decreasing rank. For the summary, we display as many words from this ordering as will fit within the word budget.

In the BP, teams were free to use their own highlighting method to display the summary. Through extensive experimentation with FigureEight[32] workers, we found that unless high- lighting was used they often labeled summaries as "Not relevant" even when the query appears in the sentence. Thus, we chose to highlight each component of a compound query with a distinct color, using green font for the first and purple font for the second. The top three most similar words for each component are printed in the corresponding font color, where similarity is determined by the cosine similarity of the embedding query and the word embedding. Exact query word matches are also highlighted in yellow, as shown in Figure 13. We augmented this by showing the top five most relevant topic words using a Latent Dirich Allocation (LDA) topic model to infer the topic of the query word from the document. We used this approach to help the Amazon Mechanical Turk (AMT) workers[33] with determining the intended meaning of the query word in context.

[32] Subsequent to this work, the company was acquired by Appen: https://appen.com/

[33] https://www.mturk.com/



```
RELATED WORDS
ISIS: syrian genocide slaughtering president al-assad
weapon: weapons used united nations months

SUMMARY
• Aleppo enters the events that commemorates human brutality , the United Nations delegate says Dec
  21 , 2016
• Then Iraqi warplanes broke out bombs with a mixture of haradali air `` mustard , '' sarin and air
  against the versus versus veins .
• In recent years the ISIS threat has increased .
• In March 1988 , during the war between Iran and Iraq , Saddam Hussein used the weapons against
  Iranian troops and civilians in Halabja town in Northern Iraq .
• The attack began using ordinary weapons against the city which

WORDS NOT FOUND: weapon
```

Figure 13. An Example Summary for a Compound Query.

A complete example output from the BP summarizer is shown in Figure 13. The related words are printed at the top of the summary, which consists of sentences that have been translated into English and have a high similarity to the query, as determined by one of the three methods described. We also included a "Words Not Found" list of the query words that were not found in the translated document.

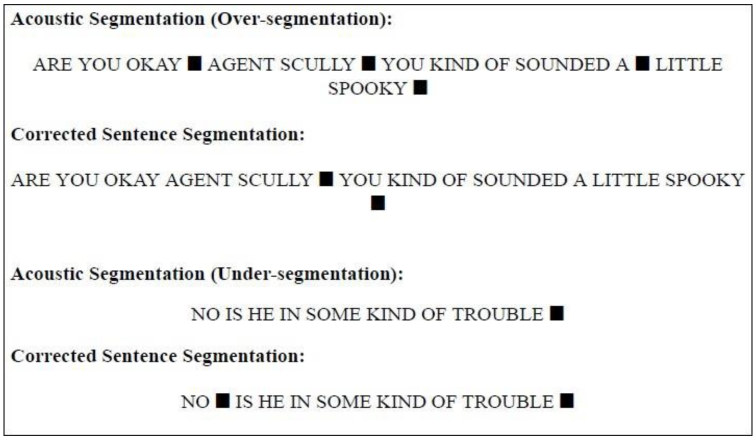
In the BP, we also did experiments to identify the most significant problems for Turkers in determining relevance, with a focus on differences between text and speech documents, as well as the fluency of the translation. Our results suggested that in OP1 we should focus on improving the fluency of speech documents and of MT in the limited context of the summary.

3.7.2 Speech Segmentation to Improve Fluency

Typical ASR systems segment audio input into *utterances* using purely acoustic information such as pauses in speaking or other dips in the audio signal. These utterances, however, may not resemble the sentence-like units that are expected by conventional MT systems for spoken language translation (SLT) [119]. In some cases, longer utterances may span multiple sentences, while shorter utterances may be sentence fragments containing only a few words, as illustrated in Figure 14. Both of these scenarios can be problematic for downstream MT systems, so we developed a model for correcting the acoustic segmentation of an ASR model to improve performance on downstream tasks, focusing specifically on the SLT pipeline challenges for LRLs.

To do this, we independently collected subtitle data and used it to train a speech segmentation model. While prior work has trained intermediate components to segment ASR output into sentence-like units [120], [121], these have primarily focused on highly resourced language pairs such as Arabic and Chinese. When working with an LRL, suitable training data may be limited to nonexistent. We therefore constructed proxy segmentation datasets using film and television subtitles, which typically contain segment boundary information like sentence-final punctuation.

Though subtitles are not exact transcriptions of audio speech, they are nonetheless closer to transcribed speech than many other large text corpora. Because subtitles still lack an existing acoustic segmentation for our model to correct, however, we generate synthetic acoustic segmentation by explicitly modeling two common error modes of ASR acoustic segmentation: under- and over-segmentation. Our model is therefore able to take as input a sequence of tokens and segmentation boundaries produced by the acoustic segmentation of the ASR system and returns a corrected segmentation.



Acoustic Segmentation (Over-segmentation):

ARE YOU OKAY ■ AGENT SCULLY ■ YOU KIND OF SOUNDED A ■ LITTLE SPOOKY ■

Corrected Sentence Segmentation:

ARE YOU OKAY AGENT SCULLY ■ YOU KIND OF SOUNDED A LITTLE SPOOKY ■

Acoustic Segmentation (Under-segmentation):

NO IS HE IN SOME KIND OF TROUBLE ■

Corrected Sentence Segmentation:

NO ■ IS HE IN SOME KIND OF TROUBLE ■

Figure 14. Example Acoustic Segmentation Errors and their Corrections.
■ *indicates a segment boundary.*

Our approach makes the following contributions:

- We propose the use of subtitles as a proxy dataset for correcting ASR acoustic segmentation.
- We develop a method for adding synthetic acoustic utterance segmentations to a subtitle dataset.
- We provide a simple neural tagging model for correcting ASR acoustic segmentation before its use in an MT pipeline.
- We investigate whether this yields downstream performance increases on MT and document-level CLIR tasks

3.7.3 Supervised Approach - Query Relevance Sentence Selection

In OP1, we moved to include a ranker using a supervised approach. By developing a simple cross-lingual embedding-based model that avoids translation entirely, our approach directly predicts the relevance of a cross-lingual query-sentence pair.

For training, we treat a sentence as relevant to a query if there exists a translation equivalent of the query in the sentence. This definition of relevance is most similar to the lexical-based relevance used in [122] and [123], but in our case, the query and sentence are from different languages.

Because we frame the task as a problem of finding sentences that are relevant to an input query, we need relevance judgments for query sentence pairs. As our focus is on LRLs, however, we lack the sentence level relevance judgments needed to train our query focused relevance model. To overcome this, we leverage noisy parallel sentence collections previously [124], [125], collected from the web. Using a simple data augmentation and negative sampling scheme, we generate a new labeled dataset of relevant and irrelevant pairs of queries and sentences from these noisy parallel corpora. We can then use this synthetic training data to learn a supervised cross-lingual embedding space.

While our approach performs comparably with the pipelines of MT with IR, it is still sensitive to noise in the parallel sentence data. Inspired by previous work in text classification that supervises attention over rationales for classification decisions [126], we find we can mitigate the negative effects of this noise if we first train a phrase-based SMT model on the same parallel sentence corpus and use the extracted word alignments for additional supervision. With these alignment hints, our system demonstrates consistent and significant improvements over neural and statistical MT+IR [127], [128], [129], as well as three strong cross-lingual embedding-based models (Bivec [130], Sentence Identification-Skip-grams with Negative Sampling (SID-SGNS) [131], Multilingual Unsupervised and Supervised Embeddings (MUSE) [132], a probabilistic occurrence model [133], and a multilingual pretrained model XLM-RoBERTa [78]. We refer to this secondary, alignment-based training objective as rationale training (RT).

Our approach for the summarizer in OP1 features:

- A data augmentation and negative sampling scheme to create a synthetic training set of cross-lingual query-sentence pairs with binary relevance judgements.
- A Supervised Embedding-based Cross-Lingual Relevance (SECLR) model trained on this data for low-resource sentence selection tasks on text and speech.
- RT secondary objective to further improve SECLR performance, which we call SECLR-with Rationale Training (SECLR-RT).
- Training data ablation and hubness studies that illustrate our method's applicability even to lower-resource settings and its ability to mitigate hubness issues [134], [135]. These findings are empirically validated by the results of our experiments in a low- resource sentence selection task, using English queries with sentence collections of text and speech in Somali, Swahili, and Tagalog as described in Section 4.6.6, Supervised Approach - Query Relevance Sentence Selection.

3.7.3.1 Training Set Generation

In this section, we describe the methods we used to create synthetic training data in order to train a supervised query relevance classification model. We describe the creation of relevant (positive) query/sentence pairs, the creation of irrelevant (negative) query/sentence pairs, the embed- ding-based query relevance model we train (SECLR), and finally our use of rationale training (SECLR-RT) to better align the embedding space and improve the performance of SECLR.

3.7.3.2 Relevant Query/Sentence Generation

For a parallel corpus of bilingual sentence pairs equivalent in meaning, let (*E, S*) be a sentence pair where *E* is in the query language (e.g., English) and *S* is in the source (e.g., LRL). For every unigram *q* in *E* that is not a stopword, we construct a positive relevant sample by viewing *q* as a query and *S* as a relevant sentence. Because sentences *E* and *S* are approximately equivalent in meaning, we know that there likely exists a translation equivalent of *q* in the sentence *S*. Thus, we label the (*q, S*) pair as relevant (i.e., $r = 1$).

An example of this can be seen with the English-Somali sentence pair: *E*="true president gaas attend meeting Copenhagen," *S*="ma runbaa madaxweyne gaas baaqday shirka copen- hegan."[34]

[34] Stopwords removed.

By extracting unigrams from *E* as queries, we generate the following positive examples: (*q*="true," *S*, *r* = 1), (*q*="president," *S*, *r* = 1), (*q*="gaas," *S*, *r* = 1), …, (*q*="Copenhagen," *S*, *r* = 1).

We generate the positive half of the training set by repeating the above process for every sentence pair in the parallel corpus. We limit model training to unigram queries, since higher- order *n*-grams appear less frequently and treating them independently reduces the risk of over-fitting. Testing shows that our model is able to process multi-word queries.

3.7.3.3 Irrelevant Query/Sentence Generation

To improve learning, we opt to also create negative examples—that is, tuples (*q, S, r* = 0)—via negative sampling. For each positive sample (*q, S, r* = 1), we randomly select another sentence pair (E', S') from the parallel corpus. We then check whether S' is relevant to *q* or not. Note that both the query *q* and sentence E' are in the same language, so checking whether *q* or a synonym can be found in E' is a monolingual task. If we can verify there is no direct match or synonym equivalent of *q* in E' then transitivity implies there is unlikely to be a translation equivalent in S'; this makes the pair (q, S') a negative example. To account for synonymy when we check for matches, we represent *q* and the words in E' with pretrained word embeddings. For example, let $w_q, w_{q'} \in \mathbb{R}^d$ be the embeddings associated with *q* and the words q' E'. We judge the pair (q, S') to be irrelevant (i.e., *r* = 0) according to Equation 20, where λ_1 is a parameter.

$$\max_{q' \in E'} \text{cos-sim}(w_q, w_{q'}) \leq \lambda_1 \tag{20}$$

We manually tuned the relevance threshold λ_1 on a small development set of query-sentence pairs randomly generated by the algorithm and set λ_1 = 0.4 to achieve highest label accuracy on the development set. If (q, S') is not relevant, we add (q, S', r = 0) to our synthetic training set. Otherwise, we resample (E', S') until a negative sample is found. We generate one negative sample for each positive sample in order to build a balanced dataset.

Building on the previous example, if we want to generate a *negative* example for the positive example (*q*="meeting," *S*="ma runbaa madaxweyne gaas baaqday shirka copenhegan," *r* =1), we randomly select another sentence pair, e.g., E'="many candidates competing elections one hopes winner," S'="musharraxiin tiro badan sidoo u tartamaysa doorashada wuxuuna mid kasta rajo qabaa guusha inay dhinaciisa ahaato," from the parallel corpus. To check whether q="meeting" is relevant to S′, it suffices to check whether q="meeting" or a synonym is present in E′, which is a simpler monolingual task. If q is irrelevant to S′, we add (q, S′, r = 0) as a negative example.

3.4.3.4 Cross-Lingual Relevance Model

SECLR is able to make direct relevance classification judgements for cross-lingual queries and sentences without relying on intermediate MT by learning a cross-lingual embedding space between the two languages. Not only should translation of equivalent words in either language be mapped to similar regions in the embedding space, but dot products between query and sentence words should be correlated with the probability of relevance.

We assume the training set generation process provides us with a corpus of *n* query-sentence pairs along with their corresponding relevance judgements, i.e. $\mathcal{D} = \{(q_i, S_i, r_i)\}|_{i=1}^{n}$.

We then construct a bilingual vocabulary $V = V_Q S$ and associate with it a matrix $W \in \mathrm{R}^{d\times|V|}$ where $wx = W\cdot,x$ is the word embedding associated with word $x \in \mathrm{V}$.

When the query is a unigram q - as in the case of our training data $\mathcal{D}'$ - we model the probability of relevance to a sentence S as shown in Equation 21, where σ denotes the logistic sigmoid *(σ(x) = 1/ (1 + exp(−x)))*.

$$p(r = 1|q, S; W) = \sigma\left(\max_{s\in S} w_q^\intercal w_s\right) \tag{21}$$

In our evaluation setting, the query is very often a phrase Q = [q1, . . . , q|Q|]. In this case, we require all query words to appear in a sentence in order for a sentence to be considered as relevant. Thus, we modify our relevance model to that shown in Equation 22.

$$p(r = 1|Q, S; W) = \sigma\left(\min_{q\in Q}\max_{s\in S} w_q^\intercal w_s\right) \tag{22}$$

Our only model parameter is the embedding matrix W, which is initialized with pretrained monolingual word embeddings and learned via minimization of the cross entropy of the relevance classification task, as shown in Equation 23.

$$\mathcal{L}_{rel} = -\log p(r|q, S; W) \tag{23}$$

As described in Section 3.7.3, Supervised Approach - Query Relevance Sentence Selection, we improve SECLR by incorporating additional alignment information as a secondary training objective, yielding SECLR-RT. Our intuition is that after training, the word $\hat{s} = \arg\max_{s\in S} w_s^\intercal w_q$ should correspond to a translation of q. It is possible, however, that $s\hat{}$ only *co-occurs* with the true translation, and the association is coincidental or irrelevant. To correct for this, we run two SMT word alignment models, GIZA++ [136] and Berkeley Aligner [137], on the original parallel sentence corpus. The two resulting alignments are concatenated, as in[2] , in order to estimate an unidirectional probabilistic word translation matrix, $A \in [0, 1]^{|^V Q| \times |VS\,|}$, such that A maps each word in the query language vocabulary to a list of document language words with different probabilities. For example, $A_{q,s}$ is the probability of translating q to s and $\sum_{s\in\mathcal{V}_S} A_{q,s} = 1$.

For each relevant training sample, i.e., (q, S, r = 1), we create a *rationale distribution* $\rho \in [0, 1]^{|S|}$. This is essentially a re-normalization of possible query translations found in S, and represents our intuitions about which words $s \in S$ that q should be most similar to in the embedding space, as shown in Equation 24, for $s \in S$.

$$\rho_s = \frac{A_{q,s}}{\sum_{s'\in S} A_{q,s'}}. \tag{24}$$

We similarly create a distribution under our model, $\alpha \in [0, 1]^{|S|}$, as shown in Equation 25, for $s \in S$.

$$\alpha_s = \frac{\exp\left(w_q^\intercal w_s\right)}{\sum_{s'\in S}\exp\left(w_q^\intercal w_{s'}\right)} \tag{25}$$

To encourage α to match ρ, we impose a Kullback–Leibler (KL) divergence penalty, to our overall loss function. This is denoted as shown in Equation 26.

$$\mathcal{L}_{rat} = \mathrm{KL}(\rho \| \alpha) \tag{26}$$

The total loss for a single positive sample then will be a weighted sum of the relevance classification objective and the KL divergence penalty as shown in Equation 27, where λ_2 is a relative weight between the classification loss and rationale similarity loss.

$$\mathcal{L} = \mathcal{L}_{rel} + \lambda_2 \mathcal{L}_{rat} \tag{27}$$

Note that we do not consider rationale loss for the following three types of samples: negative samples, positive samples where the query word is not found in the translation matrix, and positive samples where none of the translations of the query in the matrix are present in the source sentence.

3.7.4 Use of Retranslation or Automatic Post-Editing (APE)

An analysis of our prior evaluations through Farsi in OP1 showed that our biggest problem was in true detections to misses (TD-to-Miss) In other words, although CLIR identified relevant documents correctly, the summary was deemed irrelevant by AMT participants because the query word was often mistranslated, and thus did not appear in the summary. We thus looked for ways to re-translate relevant sentences that did not specifically include the query word. At the same time, we did not want to *force* the word to appear if it did not actually occur in the source.

To address this, we experimented with constrained MT as well as constrained APE, and with manipulating the conditions under which the selected sentences should be re-translated. We use PSQ to identify the sentence in the generated summaries with the highest probability of corresponding to the given query. Our reasoning was that because we only need the query word to occur once in the summary, we only needed to retranslate a single sentence.

We also found some false positive summaries, where it was not feasible for any sentence in the source to include the query word. In fact, most of our constrained MT and APE approaches were able to insert the query word into the translation even when not appropriate; this increased errors on false alarm to accept (FA-accept) We therefore developed several methods of selecting which summaries should even be considered for retranslation in the first place. For this, we used a logistic regressor that had been trained to select ground-truth relevant documents from those that had been passed to the summarization system by CLIR. Because all summaries would be presented to the end user, but only the ones selected by this regressor would be retranslated, it was appropriate for this relevance classifier to be both separate from and more stringent than the one used by CLIR. For a given document, our logistic regressor includes the following features:

- The final relevance score assigned to the document by the CLIR system (tuned for F1)
- The maximum score assigned to sentences in the document by our neural sentence relevance module
- The maximum score assigned to sentences in the document by the PSQ system used by CLIR

We trained this system on ANALYSIS (using K-Fold cross-validation) and then selected a threshold at which it would select a document to be retranslated. We tested several ways of selecting this threshold (including optimizing for F-beta with values of 0.5, 1, and 2; as well as optimizing for Receiver Operating Characteristic [ROC]); our results using AMT participants with DEV that F1 was most appropriate.

We used these results to augment the summarization approach we used for the Kazakh and Georgian evaluations in OP2. The summarization system for Kazakh and Georgian followed our approach in previous evaluations, but we added the ability to do constrained APE or an additional round of constrained MT of those sentences for which we found evidence of relevance in the source language. Specifically, we used constrained APE for Kazakh and constrained MT for Georgian.

Thus, the summarization system for our two final evaluations in OP2 consists of multiple rankers that each use the query to look for evidence of relevance in the documents retrieved by CLIR. The rankers score each sentence with its degree of relevance, and the scores from the different rankers are then merged using the Borda counts algorithm.

For Kazakh and Georgian, we used the following rankers:

1. **SLT of the query matched against the source language document.** Scores are determined using the number of PSQ matches and cosine similarity of source embeddings using Global Vectors for Word Representation (GloVe).
2. **English query matched against the English translation of the document.** Here, scores are determined using exact matches, stem matches and cosine similarity using both GloVe and Roberta embeddings. The summarization system used two different document translations produced by UMD NMT and EDIN NMT. When selecting a sentence to include in the summary, it uses the translation with the highest ranker score.
3. **English query matched directly against the source language document.** Scores are determined using a neural sentence relevance model [138], as described in Section 3.7.3. The neural sentence relevance model is our only supervised relevance system. It features three main components:
    (a) Synthetic data generation for training the system, producing English query, source sentence pairs, both positive and negative. Parallel data is drawn from Paracrawl, Wikipedia and Open Subtitles.
    (b) The model architecture, called SECLR, directly makes relevance classification judgments for queries and sentences of different languages without using MT as an intermediate step by learning a cross-lingual embedding space between the two languages. Not only should translation of equivalent words in either language be mapped to similar regions in the embedding space, but dot products between query and sentence words are correlated with the probability of relevance.
    (c) We augment the relevance model using additional alignment information as a secondary training objective, obtained through PSQ. We use KL-divergence between the two training objectives (relevance and alignment) when training the model. More details can be found in [138].

3.7.4.1 APE

APE aims to improve the quality of the output of an arbitrary MT system by pruning systematic errors and adapting to a domain-specific style and vocabulary [139], [140]. Although previous work has shown the usefulness of APE to prune errors by focusing on improving the translation error rate (TER), few have studied the effect of incorporating lexical constraints.

For the MATERIAL use case, lexically constrained APE is useful for CLIR. When displaying snippets from retrieved documents, the query term should appear in the translation output if it does in the source, in order to make relevance clear to the human end user. We note that such a system would also be beneficial in a range of other contexts; for example, content providers often meticulously curate lists of terminologies for their domains that indicate preferred translations for technical terms.

While recent approaches support inference time adaptation of NMT systems using manually curated term lists, post-editing translations with a generic APE system may lead to dropped terms. A constraint-aware APE system would make it possible to fix systematic translation errors, while still keeping the terms intact.

We consider a range of representations that augment input sequences with constraint tokens and factors for use in an Autoregressive Transformer (AT) APE model. Using this approach, the constraints were explicitly represented in the encoder input sequence and the model learns to prefer translations that contain the supplied terms during decoding. We also explore the use of the Levenshtein Transformer (LevT), a Non-Autoregressive Transformer (NAT) model. The LevT model applies neatly to the APE task since the decoder can be initialized with an incomplete sequence. Additionally, multiple corrections can be made simultaneously, resulting in faster decoding than autoregressive models.

We experiment extensively with variations of both AT and LevT models, testing on both phrase-based machine translation (PBMT) and NMT English to German WMT APE tasks [140]. Under all scenarios, the model performs post-editing while satisfying terminology constraints when supplied.

Using constraints constructed with synonyms and antonyms, we also investigate whether our models learn to copy constraints systematically and introduce a simple data augmentation strategy [141] to improve the preservation of unusual constraints.

To summarize, our approach features:

- AT and LevT model variants for incorporating lexical constraints.
- An investigation into whether constrained APE is necessary to preserve terminology constraints in a MT to APE pipeline.
- An analysis of the robustness of the constraint translation behavior and a simple data augmentation technique that improves translation quality and increases the number of correctly translated terms.

3.7.5 Other Experimental Approaches

Throughout this work, we explored a number of new approaches that were not adopted in the final system. For example, we observed that the fluency of a summary seemed to affect whether or not it was selected by AMT workers and reasoned that we might be able to improve the fluency of the summary translation using abstractive summarization. Because the summary is not the entire document, an abstractive approach could feasibly reword only those sentences needed for fluency, in contrast with an extractive method that would need to make a trade-off between conciseness and fluency. We first published our approach to generic abstractive cross-lingual summarization in the 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL) 2019 [142] and have since extended that work for the task of query- focused abstractive summarization. We also experiment with methods to improve accuracy on conceptual queries; this work was concluded after the last evaluation.

3.7.5.1 Abstractive Cross-lingual Summarization

Cross-lingual summarization is a little explored task that combines the difficulties of automatic summarization with those of MT. The goal is to summarize in one language a document available only in another language. [143] describe two reciprocal approaches to this task: summarize then translate and translate then summarize. They argue that summarize-then-translate is preferable, as it minimizes the computational expense of translating more sentences as well as the sentence extraction errors caused by incorrect translations. Summarize-then- translate is only effective for high resource languages, however, when working with LRL, no summarization corpora are available to support this approach. Language-independent techniques, such as TextRank (Mihalcea) [144], might be used, but morphologically rich languages render such token-based similarity measures useless. In these instances, translate-then-summarize is the only possible approach.

To address this problem, we develop a neural abstractive summarization system that fluently summarizes automatically-translated documents by generating short, simple phrases to replace any awkward input phrases. Our novel recombination of existing systems results in a summarization solution that can be easily applied to new LRLs. In this approach, we use MT on the *NYT* annotated corpus of document/summary pairs [145] to create summarization corpora for documents automatically translated from three LRLs: Somali, Swahili, and Tagalog. We begin by translating the *NYT* documents to Somali, Swahili and Tagalog, and then re-translating them back to English. This yields the type of disfluent input documents we might expect from a translation system; then we pair these with the fluent English summaries from the original corpus.

We use these corpora to train cross-lingual summarizers for these source languages, with English as the target. We also evaluate our systems on a fourth source language - Arabic. We investigated whether our abstractive summarizers produced more fluent English summaries from automatically-translated documents and whether this improvement generalizes across source languages.

Our approach features:

- The creation of a summarization corpora for automatically translated Somali, Swahili, and Tagalog documents
- The creation of noisy English input documents paired with clean English reference summaries
- A method for producing cross-lingual summarization systems for LRL where no summarization corpora currently exist, thereby providing a potential summarization solution for thousands of such languages
- Assessments of whether our novel approach outperforms a standard copy-attention abstractive summarizer on real-world LRL documents in Somali, Swahili, and Tagalog, as well as whether our approach generalizes to unseen languages, tested using a set of Arabic documents paired with English summaries created for an earlier Document Understanding Conference

(DUC) evaluation.

3.7.5.2 Conceptual Query Processing

A crucial component of summarization is choosing the most relevant sentences to display. While it is easier to match sentences relevant to lexical queries by considering the occurrences of the query word in a sentence, retrieving correct sentences for conceptual queries requires semantic knowledge and more sophisticated methods. This work follows the success of SECLR-RT [146] in generating a synthetic similarity dataset using parallel corpora by learning a model that maps the embedding space of English lexical queries with sentences in the source language. While SECLR-RT shows promising results on lexical queries, the difficulty of finding relevant sentences for conceptual queries was not addressed. Thus, we intend to leverage Transformer-based contextualized embeddings for their semantic understanding, along with multilingual training, to improve performance on languages with less training data.

To avoid errors of translation affecting our matching process, the model performs semantic matching directly on the source sentence and the English queries. Because suitable training data for such tasks does not exist, we create synthetic training data by using parallel data and then generate queries similar to those in evaluation by extracting phrases from the English sentence. Finally, we explore various training tasks to learn a mapping of sentences from multiple languages into the same embedding space.

3.8 Integration

Most components of the SCRIPTS systems are complete and self-contained, meaning they can be run selectively and independently of one another. This means that users of SCRIPTS can choose to run the individual components of the overall system (e.g., ASR, MT, morphological analyzer, matchers, etc.) manually with the option to validate, reformat or otherwise modify the outputs and/or inputs of each step.

In order to streamline use of the overall system and increase efficiency, however, we have built an executive component in Java that automates the sequential, E2E SCRIPTS input/output processes. This executive component allows a user to generate document matching, summarization, and all other intermediate SCRIPTS outputs without learning how to run individual sub components.

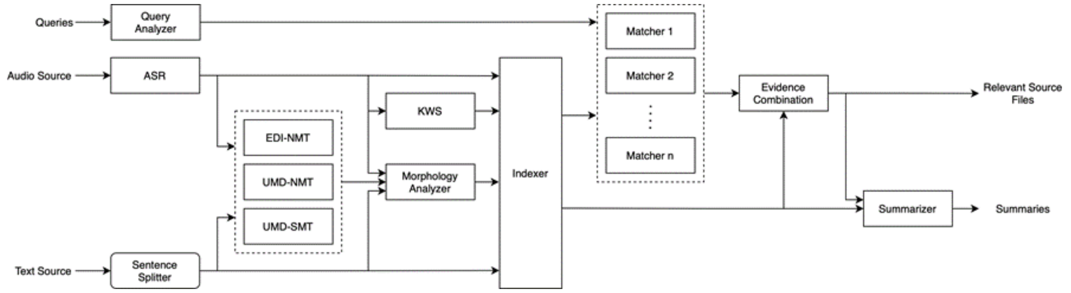


Figure 15. Overview of the SCRIPTS System as Implemented by the Executive Component.

All SCRIPTS components - including the executive component - are provided as Docker images, and therefore require a Docker daemon to run. Once all the component Docker images have been installed on the destination machine per the requirements and directions provided in the SCRIPTS pipeline documentation,[35/36] and the system completeness is confirmed, it is available to process data. The required inputs include:

- audio source documents (.wav files)
- text source documents (.txt files)
- a list of queries (.tsv file)
- a set of configuration files (.xxx files)

Once the executive component has been initiated, it will reformat and relocate the input as needed and will launch the dockerized sub-systems (e.g., ASR, MT, morphological analyzer, matchers, etc.) in the predefined order. Running on an Amazon Web Service (AWS) p3.8xlarge instance, the system currently needs approximately five days to completely process a typical data set of 3,000 audio files, 10,000 text files and 1,200 queries – though actual run time will also depend significantly on factors such as source language, query complexity, and the size of the input files.

There are several design choices that were made for the executive component in order to make the system as efficient as possible. One of the design focuses was to make the system modular. The system takes in a configuration file where users can specify the versions of the components to be used by the system. The executive component also invokes other components via separate shell scripts instead of hard coding it into the Java application. This design allows the user to update docker components with a different version or component run scripts without recompiling the whole application.

The executive component also saves intermediate outputs to disk, which allows the users to choose to run the system up until a specific point and then continue running the rest of the system later or the user could branch out from a point and try a new experiment with a different configuration easily. Files do not expire and access is shared across all users. If a component exists, it will not be re-processed.

[35] https://github.com/hl3436/material-scripts-pipeline
[36] https://github.com/hl3436/material-scripts-pipeline/tree/main/example/workspace-fa

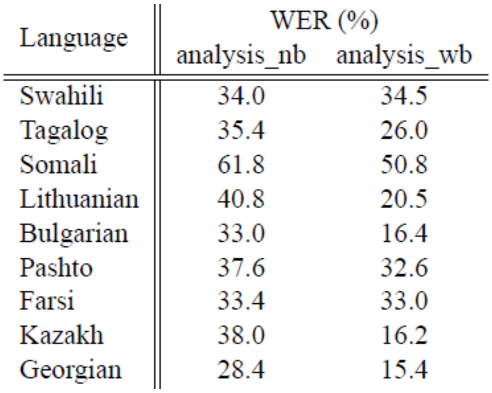
4.0 PROGRAM RESULTS, FINDINGS AND TECHNICAL IN- SIGHTS

4.1 Speech Processing

4.1.1 Baseline Systems and Diarization

Table 3 contains the final WERs achieved with the dockerized systems for all languages, each of which was built using a combination of the baseline pipeline described in Section 3.1.1 and the SST process detailed in Section 3.1.3.

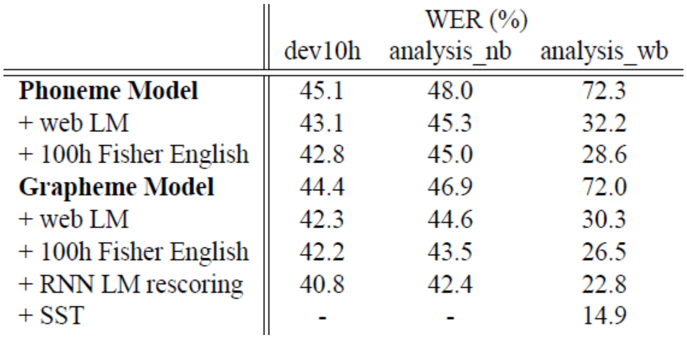
Table 3. WERs % for each Language of Dockerized Final Single Systems.



Language	WER (%)	
	analysis_nb	analysis_wb
Swahili	34.0	34.5
Tagalog	35.4	26.0
Somali	61.8	50.8
Lithuanian	40.8	20.5
Bulgarian	33.0	16.4
Pashto	37.6	32.6
Farsi	33.4	33.0
Kazakh	38.0	16.2
Georgian	28.4	15.4

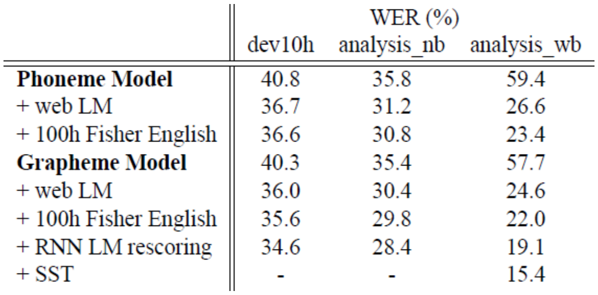
Tables 4 and 5 illustrate how specific properties of, or augmentations made to, our baseline systems impacted performance for our Kazakh and Georgian system results in particular, though we note that the most successful methods/system features described here were also generally successful for the remaining languages.

Table 4. Comparing the WER % Achieved by Various EDIN Systems for Kazakh.



	WER (%)		
	dev10h	analysis_nb	analysis_wb
Phoneme Model	45.1	48.0	72.3
+ web LM	43.1	45.3	32.2
+ 100h Fisher English	42.8	45.0	28.6
Grapheme Model	44.4	46.9	72.0
+ web LM	42.3	44.6	30.3
+ 100h Fisher English	42.2	43.5	26.5
+ RNN LM rescoring	40.8	42.4	22.8
+ SST	-	-	14.9

Table 5. Comparing the WER % Achieved by various EDIN Systems for Georgian.



	WER (%)		
	dev10h	analysis_nb	analysis_wb
Phoneme Model	40.8	35.8	59.4
+ web LM	36.7	31.2	26.6
+ 100h Fisher English	36.6	30.8	23.4
Grapheme Model	40.3	35.4	57.7
+ web LM	36.0	30.4	24.6
+ 100h Fisher English	35.6	29.8	22.0
+ RNN LM rescoring	34.6	28.4	19.1
+ SST	-	-	15.4

4.1.1.1 Impact of Adding Web-Scraped Data on WERs

In general, adding scraped, external data to the provided build sets improved performance in terms of WER. For acoustic data, adding the 100h Fisher English data set to the provided build CTS data led to marginal WER improvements for both languages, indicating the potential utility of OOL resources. Similarly, augmenting the build data with the scraped data improved the LM for both languages, leading to reduced WERs, especially for the analysis_wb set. Adding LM rescoring to the EDIN pipeline (using a neural RNN LM) enabled WER% to be reduced still further.

4.1.1.2 Impact of Adding Grapheme-Based Models

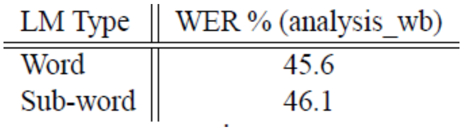
As mentioned in Section 3.1.1, the SCRIPTS team also evaluated phoneme versus grapheme-based models. As shown in Table 4 and Table 5, the grapheme-based models and lexicons proved superior for both the languages shown here and across the board. As a result, we chose to use grapheme-based models as seeds when conducting subsequent SST for all OP1 and OP2 languages, as described in Section 4.1.3.

The success of our grapheme-based modelling approach also allowed us to briefly experiment with LM at the sub word rather than word level, which hypothetically could help alleviate OOV or data sparsity issues - especially for languages with a more complex or agglutinative morphology (e.g., Georgian). In practice, however, our limited experiments using this approach with the Pashto language proved inferior. The results of these explorations can be seen in Table 6, which compares Pashto systems using sub word level versus word level LMs. Still, it is possible that different sub word model configurations and/or hyper-parameter values might yield better results than those obtained here.

4.1.1.3 Hybrid and E2E Systems

Table 7 shows a contrast, using exactly the same training data, between "traditional" hybrid system and encode-decoder attention-based model, so called E2E model. In terms of WER on Technology, Entertainment, Design (TED) - Laboratoire d'Informatique de l'Université du Maine (LIUM) and the Kazakh WB data the hybrid system outperforms the E2E system. It should be emphasized these systems were trained using the exact same quantity of training data. E2E systems may be expected to benefit more from, for example, transfer learning but this has not been explored by the SCRIPTS team.

Table 6. Effect of Word vs Sub-word LM when Building an ASR System for Pashto (Note: Both Systems here used Common-Crawl data for LM training); Time Delay Neural Networks (TDNN) Factored Form-(F) AMs; and Additional Backus Naur Form Notation (BNF).



LM Type	WER % (analysis_wb)
Word	45.6
Sub-word	46.1

Table 7. Performance (% WER) Comparison (Kazakh) of Encoder-Decoder Attention-based (E2E) Models and Baseline Hybrid Models.

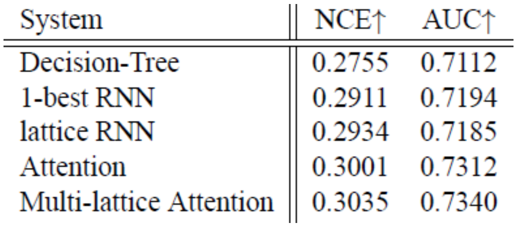
Task	Hybrid	E2E
TED-LIUM	6.9	8.2
Kazakh (WB)	17.5	18.7

4.1.1.4 Impact of DL Approaches for Confidence Score Estimation

Table 8 shows the impact of using the rich information stored on each arc of a lattice, including acoustic and LM scores, arc posterior, duration, word embedding (identity), and (optionally) the phone or character sequence embedding from the word along with durations.

Two standard confidence performance metrics were used to evaluate performance: Normalized Cross Entropy (NCE); and Area under the Curve (AUC) for precision and recall detecting incorrectly recognized words. The results of the baseline approach - which uses word posteriors and decision trees to calibrate the confidence scores - is shown in the first line. The performance improves as increasingly complex implementations of DL are used to combine the information associated with each of the arcs, with the best performance resulting from applying attention- based mechanisms using information from multiple lattices. For this multiple lattice system, the lattices were generated by using different LM scale factors. It is then possible to combine information both from within, and between the lattices using attention mechanisms. The attention mechanism here uses the arc for which we require the confidence score as the key to compute the appropriate attention weights across the lattices.

Table 8. Impact of Confidence Score Approach Georgian CTS Task.



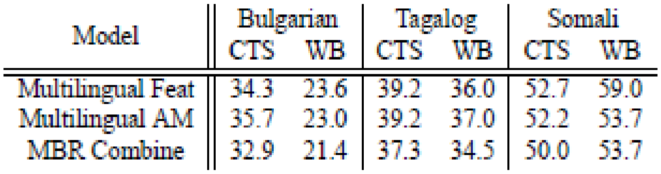
System	NCE↑	AUC↑
Decision-Tree	0.2755	0.7112
1-best RNN	0.2911	0.7194
lattice RNN	0.2934	0.7185
Attention	0.3001	0.7312
Multi-lattice Attention	0.3035	0.7340

Though these DL approaches yield performance gains in terms of the accuracy of the confidence scores obtained from the NCE score and the rank ordering from the AUC score, the relative improvement on CLIR performance in initial experiments was small compared to the baseline decision-tree based approach. We posit that this is due to the fact current CLIR performance is dominated by the probabilities associated with the translation table, rather that the ASR confidence scores.

4.1.2 Multilingual AMs

For BP and OP1, we investigated the value of adding information from multiple languages for training the initial CTS NB AM. As shown in Table 9, adding multilingual information - both in the form of bottleneck multilingual features and the "hat-swapping" framework described in Section 3.1.2, yielded performance gains on both the CTS and WB data. Though neither a feature-based nor a model-based approach was consistently better, combining the two generated performance gains for all languages.

Table 9. Performance (% WER) Comparison of Hybrid Multilingual Features and Multilingual AMs.



Model	Bulgarian		Tagalog		Somali	
	CTS	WB	CTS	WB	CTS	WB
Multilingual Feat	34.3	23.6	39.2	36.0	52.7	59.0
Multilingual AM	35.7	23.0	39.2	37.0	52.2	53.7
MBR Combine	32.9	21.4	37.3	34.5	50.0	53.7

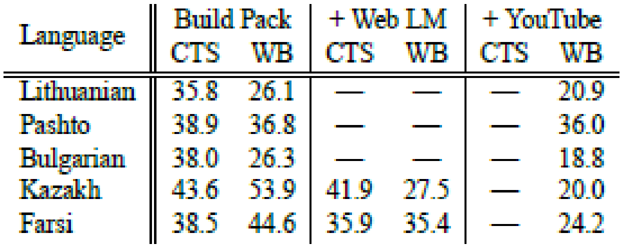
These multilingual approaches were not used for the OP2 phase of the project, as the performance gains for CLIR - especially when using large quantities of untranscribed WB audio data and text LM data that had been scraped from the web - made the multilingual optimizations less relevant.

4.1.3 Use of Web-crawled Data and SST

The standard approaches for developing ASR systems by optimizing performance on a development set cannot be used if there are no transcribed data in the target domain. Because using WER as the performance metric also requires transcribed data, where no transcribed data are available, the average confidence score of untranscribed held out data can be used instead. Figure 16 shows the relationship between the average development set confidence score and actual WER on that data on WB and NB (CTS) data from the Babel and MATERIAL programs, illustrating the strong (negative) correlation between the WER and confidence score. These results highlight the possibility of doing system development without transcribed data. This approach was not ultimately used for the MATERIAL evaluation system, as suitable transcribed training data had been made available for the target domains.

Web-scraped text data, and semi-supervised learning for AMs, was applied for all languages by the SCRIPTS team. Table 10 shows the performance of the baseline confidence-based data selection approach on a range of languages. Considering the Kazakh results, it is clear the addition of web-scrapped text data for the LM (+Web LM) gives considerable gains for the WB data, 53.9% to 27.5%, and only small gains for the CTS data. This is not surprising as the seed model was only trained on CTS style data. Additionally, building semi-supervised AMs on the YouTube scrapped data (+YouTube) gives further significant gains reducing the WER from 27.5% to 20.0%.

Table 10. BP Language Verification (Recognition) Results.



Language	Build Pack		+ Web LM		+ YouTube	
	CTS	WB	CTS	WB	CTS	WB
Lithuanian	35.8	26.1	—	—	—	20.9
Pashto	38.9	36.8	—	—	—	36.0
Bulgarian	38.0	26.3	—	—	—	18.8
Kazakh	43.6	53.9	41.9	27.5	—	20.0
Farsi	38.5	44.6	35.9	35.4	—	24.2

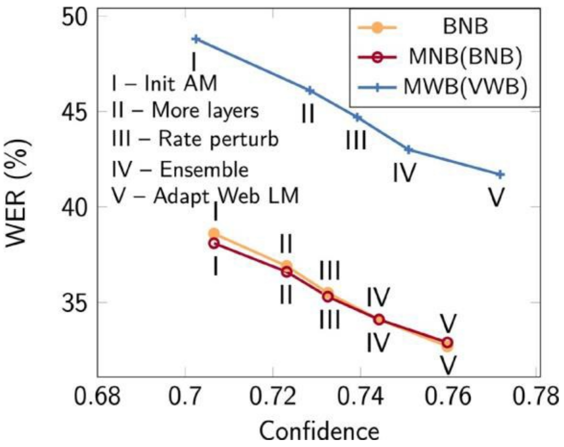
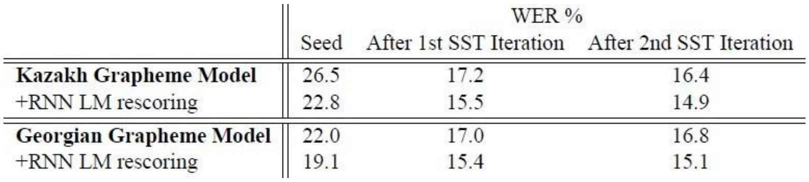


Figure 16. Confidence Scores and WER for Systems Evaluated on MATERIAL WB (MWB), Babel NB (BNB) and MATERIAL NB (MNB).

Expanding on results from Table 4 and Table 5, Table 11 further elaborates the impact of adding SST to EDIN's baseline ASR pipeline. While we once again constrain our detailed results to Kazakh and Georgian, the SST approach was successful across all languages.

Both Kazakh and Georgian saw a significant (minimum 5% absolute) drop in the WER when comparing the seed model (which was trained on just the initial build and English data) and the revised model produced after just one iteration of SST using 'pseudo-labelled' YouTube data. These results highlight the remarkable usefulness of SST in low resource scenarios, where it enables large quantities of web-scraped, untranscribed audio data (when processed using appropriate audio filtering techniques) to be successfully integrated into training - despite the large amounts of noise such data is likely to contain. In the case of Kazakh, an additional round of incremental SST lowered the WER further still, leading to an overall WER reduction of 10.1% as compared to the baseline seed model for this language.

Table 11. WER Before and After Applying SST (Using 800 Additional Hours of Scraped Data) to EDIN Systems (baseline+webLM+Fisher 100h English Data).
Evaluation conducted on analysis_wb dataset.



	WER %		
	Seed	After 1st SST Iteration	After 2nd SST Iteration
Kazakh Grapheme Model	26.5	17.2	16.4
+RNN LM rescoring	22.8	15.5	14.9
Georgian Grapheme Model	22.0	17.0	16.8
+RNN LM rescoring	19.1	15.4	15.1

The success of the SST approach led us to conduct further experiments in order to more precisely define the learning dynamics of SST for low resource/out-of-domain ASR. Specifically, we investigated two independent possibilities: how SST success is affected by the quality of the seed AM and how the quality of the LM utilized for decoding the semi-supervised data impacted the results. The intuition behind selecting these approaches was that a good quality LM may prove to be the more important of the two, because it is LM which provides external information to the system (see discussion in Section 3.1.3) [23]. Given EDIN's hybrid-based setup (described in Section 3.1.1), it was possible to isolate the AM and LM components and thus independently switch each one out for artificially degraded variants trained on less data, while still keeping the other constant. This allowed us to chart how any given AM/LM combination affected the ‘gains’ provided by SST. Figure 17 plots the initial WER achieved by the seed system against that achieved after pseudo-labelling/retraining using extra data from one such experiment conducted on Tagalog [23].

The blue series in Figure 17 demonstrates the gains that are made (illustrated by the points lying below $y = x$) when a high quality LM is combined even with progressively weaker AMs (which have increasingly high initial WERs) and applied to the semi-supervised data. Not only are such gains were made across the board, they can be observed even when the initial system’s WER was over 80%. By contrast, the orange series illustrates that even a good quality AM cannot similarly compensate for successively weaker LM variants. Though improving LM quality led to increased the SST pay-off, a poor LM could not be salvaged by a high quality AM.

These findings further demonstrate the value of SST for low resource ASR, as it supports significant WER gains even when acoustic data is extremely limited - as long as there is sufficient *text* data available to train a good quality LM.

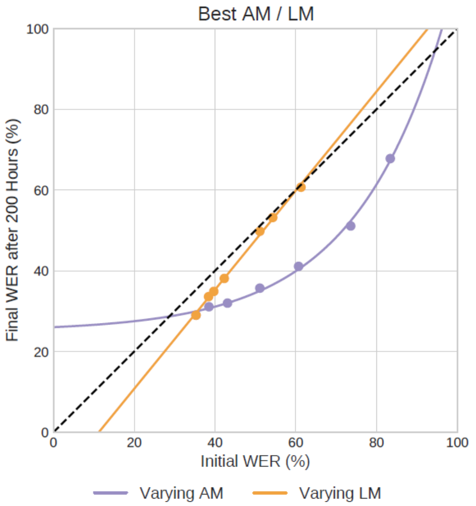


Figure 17. How Varying the Seed AM (purple) and LM (orange) Affects WER Gains Made during SST in Tagalog.

4.1.4 Integration with Downstream Tasks - Audio Document Language Verification

During the BP, the teams were required to ensure no hits were returned for audio documents that were not in the target language. To achieve this, SCRIPTS relied on document-level confidence scores to identify these OOL documents. Figure 18 shows the performance of document (speaker level) confidence scores for the target language Swahili, meaning that all confidence scores are derived from a Swahili ASR system. In both plots, the cumulative plots of documents (speakers) against confidence scores shows that the confidence scores for Swahili are significantly higher than those for both geographically close languages (left hand plot) and a broader range (right hand plot) of languages.

For the BP evaluation of language identification, a threshold was applied to the confidence scores (the equivalent of applying a vertical line in Figure 18). Documents to the right of that line are labeled as being in the target language, those to the left as not in the target language. The performance of this classifier on the evaluation target language of Somali is shown in Table 12.

As expected from the confidence distribution plots, the performance is very high.

4.1.4.1 Integration with CLIR

The simplest method of integrating ASR with CLIR is to pass the one best output to CLIR, which can then treat the audio document as a text document. For low resource speech recognition, however, here will be errors in the ASR transcriptions that will degrade CLIR performance, especially for phrases. While including confidence scores in the CLIR process is both simple and effective, however, it does not make maximum use of the output of the ASR system.

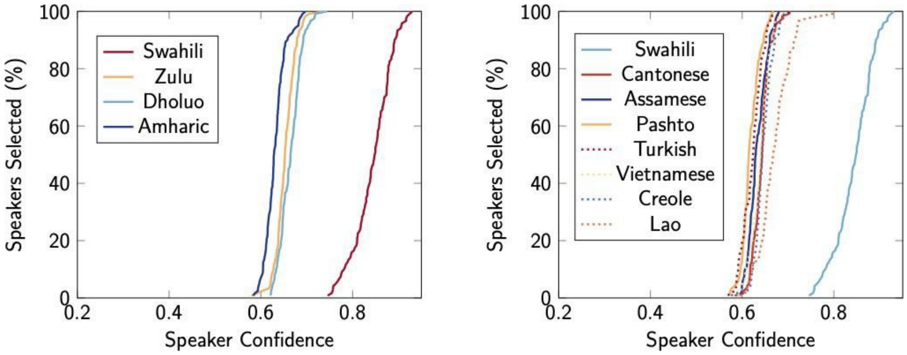
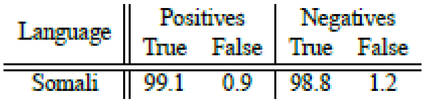


Figure 18. Confidence-Based Language Verification for Swahili: Percentage of Documents (Speakers) below a Confidence Threshold.

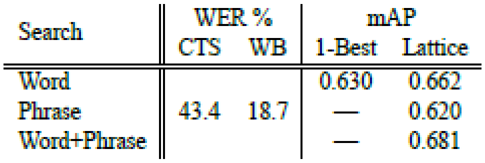
Table 12. BP Language Verification (Recognition) Results.



Language	Positives		Negatives	
	True	False	True	False
Somali	99.1	0.9	98.8	1.2

By using lattices, rather than the one best output (including confidence scores), the MAP performance can be improved significantly. For example, Table 13 shows the impact for Kazakh, where the MAP performance improves from 0.630 to 0.662. As discussed in Section 3.1.4, it is possible to use both phrase and word indices. Implementing this directly - and only considering the longest phrases from the phrase table in the English query phrase - the performance was degraded slightly to 0.620. We believe this degradation was due to a combination of less accurate translation table probabilities for phrases, and phrases being missed from the lattices.

Table 13. Performance (MAP) for Different CLIR Search Strategies on Kazakh.



Search	WER %		mAP	
	CTS	WB	1-Best	Lattice
Word	43.4	18.7	0.630	0.662
Phrase			—	0.620
Word+Phrase			—	0.681

As discussed in Section 3.1.4, it is possible to generate a single index that supports both word and phrase search by combining the results of both the word and phrase decompositions. Rather than directly combining all the probabilities, however, a weighted combination was used. Figure 19 shows the impact of different weights on the MAP performance. As expected, the weights for the word-based decomposition is higher, reflecting the more accurate translation table probabilities and higher chance of occurring in the ASR lattice. This provided an additional performance gain over the word lattice search.

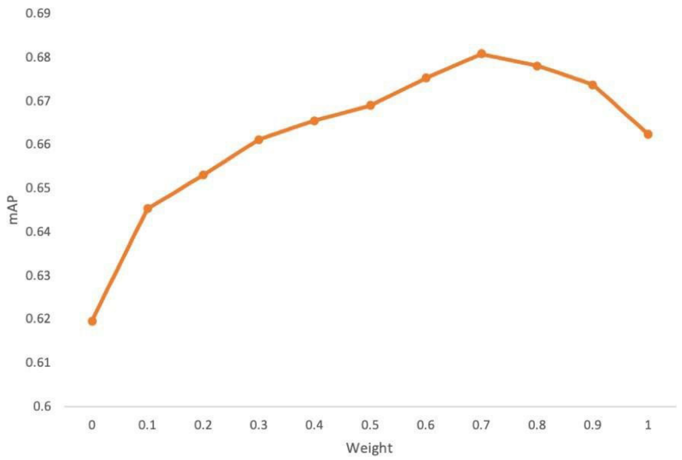


Figure 19. Word and Phrase Based CLIR Interpolation MAP Performance Kazakh, Weight Indicates the Weight Given to the Word Probabilities.

4.1.5 Technical Insights

Our results from this work indicate some clear insights with respect to low resource speech recognition:

- Web-scraped data, both audio and text, is very effective in building high performance ASR systems.
- Appropriately handling errors in the transcriptions of untranscribed web-crawled data is important. Approaches examined here included the use of confidence scores, incremental transcription, and lattices.
- Cross-domain porting—for example, CTS data to broadcast-style data - can be made possible with the use of untranscribed web-scraped data.
- If resources are not highly constrained, confidence scores from an ASR system are a simple and effective way of detecting whether the audio document was in the target language of interest.
- Appropriate filtering of untranscribed audio data at the video level (e.g., via speaking rate and first pass confidence scores, as discussed in Section 3.1.3) is also important when trying to utilize SST successfully.
- It is important to deal with error propagation from the ASR system to CLIR. Here we used approaches based on confidence scores and lattices for tighter integration.
- SST techniques can be used to greatly improve performance, even when there is very little supervised acoustic data initially available - as long as enough text data can be sourced to train a sufficiently high quality LM.

4.2 MT

Intrinsic evaluation of MT quality shows that NMT systems outperform SMT systems for all languages of the program, including in the lowest resource settings. However, downstream paths benefit from access to a diverse set of translation approaches. In particular, SMT systems are sometimes more useful for CLIR than neural systems. We also show that novel neural architectures can successfully incorporate terminology constraints in translation, which is useful to encourage MT systems to use query words when translating relevant documents.

4.2.1 Results and Findings – Edinburgh Machine Translation (EDINMT)

In addition to the other languages in the program, the provided EDINMT docker container supports translation between Kazakh-English (kk-en) and Georgian-English (ka-en) language pairs.

For each kk-en and ka-en, three models are provided:[37] (1) a text translation model; (2) a text translation model that incorporates query terms; and (3) a translation model that has been optimized for use with ASR outputs through lowercasing and removal of punctuation on the source side. Each model can be run in two modes: "accurate," which uses an ensemble of four separately trained models for the language direction, and a "fast" mode which uses only the best model.

Translation can be performed in 1-best mode, *n*-best mode or in *n*-best-words mode. In 1-best mode, the system outputs the top hypothesis based on beam search; in *n*-best mode, it outputs the top *n* sentence hypotheses. In *n*-best-words mode, the system outputs *n* tokens for each position in the top beam search hypothesis.

The MT system for Kazakh in OP2 follows the approach in previous evaluations, with the addition of a model trained specifically to incorporate query terms into the MT output that are relevant in the source language.

When training the query system, we augment our parallel data with target query terms. Spans of target text between one and three words in length are extracted from the target side parallel sentences and appended to the source side with the delimiter character |||. We append a term with 75% probability. With 25% probability, we use no augmentation in order to ensure that the model is still capable of making translations when a query term is not provided. The model is otherwise trained as per the normal method.

During inference, a user can select either the query model or the normal model, both of which were available in the final system. When using the query model, the user can optionally provide the system with a source sentence and a target term, which is then appended to the source side, in order to guide the model into incorporating that term in its output.[38] During development, the Kazakh systems were evaluated on the WMT2019 newstest set for kk-en using SacreBLEU score.

The MT system for Georgian follows our approach in previous evaluations, with two changes. First, we use SentencePiece[39] on its own for preprocessing our data with BPE. We set the vocabulary to 32k tokens, and included the numerals 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 in the set of user-defined symbols. This ensures that numerals are always segmented into their own tokens, which aids the translation of numbers. Second, we add an additional step for handling artifacts such as,

[37] For English to Kazakh/Georgian we provided only the text translation model

[38] Note that a query system was trained only for the Kazakh to English direction.

[39] https://github.com/google/sentencepiece

URLs, emails, and Extensible Markup Language (XML) tags in the text. For training the system a regular expression is run over the data to extract these artifacts and replace them with a special numbered token, such as [URL], [URL1], [URL2], etc.; these are also added to the Sentence-Piece BPE model as user defined symbols. In the translation data, artifacts are replaced by a token with 80% probability, while with 20% probability, the artifact is not replaced, so that the model can still learn to translate in case these artifacts exist in the text. The number of the token to insert is incremented from 1 to 10 (and then starts over again at number 1). This is done to allow the model to see tokens of higher numbers more often than it otherwise would. Thus, for sentences where there is only one artifact, the model won't always be exposed to only [URL1], but will also occasionally see [URL2], [URL3], etc., up to [URL10].

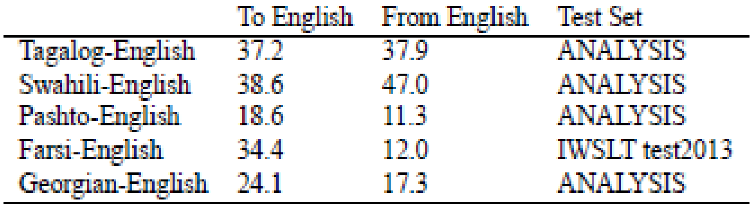
During inference, the data are similarly preprocessed except that the extracted artifacts are temporarily stored and then reinserted on the output side. In case the MT model fails to output the required token, the artifact is simply appended to the end of the sentence. During development, the Georgian systems were evaluated using Bilingual Evaluation Understudy Score (BLEU) on held out ANALYSIS data.

We have shown the final results of EDINMT systems in Table 14 and 15. Note that the Farsi-English systems were evaluated on International Workshop on Spoken Language Translation (IWSLT) test sets. For the Kazakh-English pair, we didn't have enough direct parallel data. So we used additional parallel data through a pivot language, Russian. We explored different techniques such as pivot-based NMT, multilingual training, and back-translation. The results are shown in Table 15.

4.2.2 Results and Findings - UMD NMT

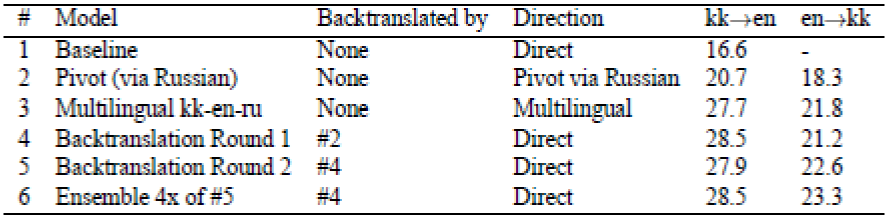
We measure translation quality using case insensitive BLEU computed by SacreBLEU [147].[40] Figure 20 shows the ablation study on the impact of different components on English↔Kazakh

Table 14. BLEU Scores of the Final EDINMT Systems.



	To English	From English	Test Set
Tagalog-English	37.2	37.9	ANALYSIS
Swahili-English	38.6	47.0	ANALYSIS
Pashto-English	18.6	11.3	ANALYSIS
Farsi-English	34.4	12.0	IWSLT test2013
Georgian-English	24.1	17.3	ANALYSIS

Table 15. BLEU Scores of Different Systems for Kazakh-English Pair on WMT2019 Newstest Set.



#	Model	Backtranslated by	Direction	kk→en	en→kk
1	Baseline	None	Direct	16.6	-
2	Pivot (via Russian)	None	Pivot via Russian	20.7	18.3
3	Multilingual kk-en-ru	None	Multilingual	27.7	21.8
4	Backtranslation Round 1	#2	Direct	28.5	21.2
5	Backtranslation Round 2	#4	Direct	27.9	22.6
6	Ensemble 4x of #5	#4	Direct	28.5	23.3

[40] Version string: BLEU+case.lc+numrefs.1+smooth.exp+tok.13a+version.1.2.11

- **Transformer vs. LSTM:** Transformer outperforms LSTM by +4.2 and +3.8 BLEU on English Kazakh and Kazakh English, respectively.
- **Impact of Back Translation:** Incorporating English monolingual data via back translation improves BLEU by +1.7 on Kazakh English.
- **Impact of Pivot-based Data Augmentation:** Pivot-based data augmentation further improves BLEU by +1.2-1.4.
- **Impact of Ensemble Decoding:** Ensembling four models further improves over singlemodels by +0.8-0.9 BLEU.

4.2.3 Results and Findings – SMT vs NMT

SMT vs. NMT Table 16 shows a comparison between NMT and SMT based on BLEU and CLIR performance measured by MQWV on Farsi-English. Results suggest that, although NMT achieves substantially higher BLEU than SMT, SMT is still useful for CLIR as it leads to better CLIR performance.

Table 16. SMT vs NMT: BLEU and CLIR Performance Measured by MQWV on Farsi-English.

	BLEU	MQWV
NMT	23.59	0.227
SMT	16.74	0.386

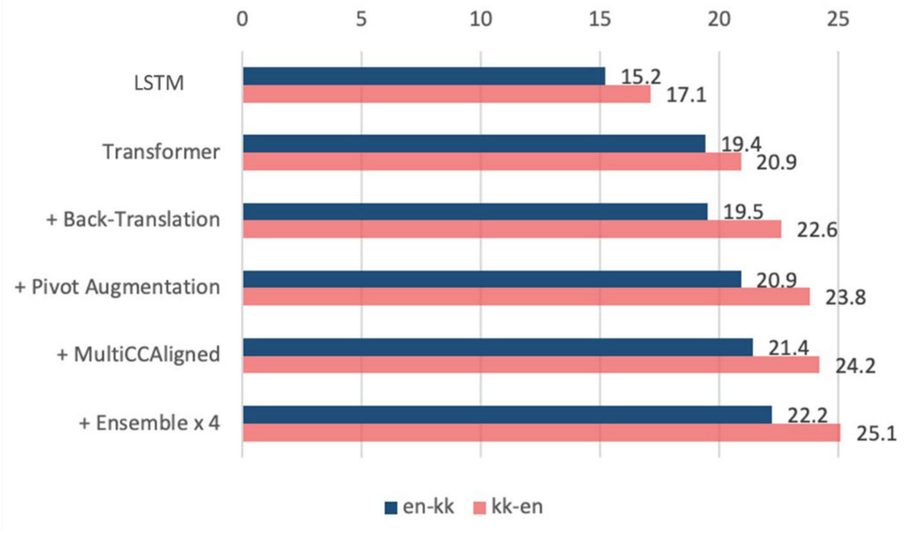
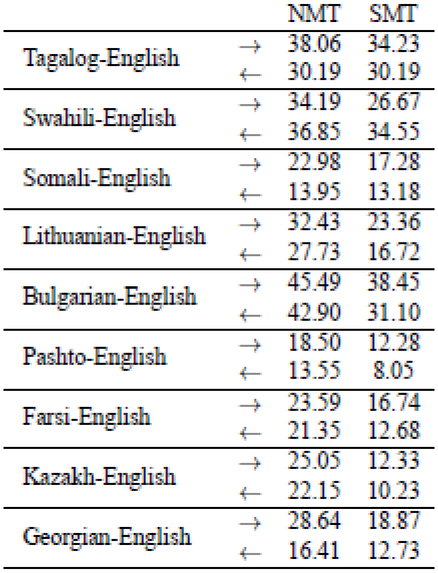


Figure 20. BLEU Scores on English↔Kazakh ANALYSIS Test Set.

4.2.4 Overall Results

Table 17 shows the BLEU scores of the final NMT and SMT systems on ANALYSIS test set.

Table 17. BLEU Scores of the Final UMD NMT and SMT Systems on ANALYSIS Test Set.



		NMT	SMT
Tagalog-English	→	38.06	34.23
	←	30.19	30.19
Swahili-English	→	34.19	26.67
	←	36.85	34.55
Somali-English	→	22.98	17.28
	←	13.95	13.18
Lithuanian-English	→	32.43	23.36
	←	27.73	16.72
Bulgarian-English	→	45.49	38.45
	←	42.90	31.10
Pashto-English	→	18.50	12.28
	←	13.55	8.05
Farsi-English	→	23.59	16.74
	←	21.35	12.68
Kazakh-English	→	25.05	12.33
	←	22.15	10.23
Georgian-English	→	28.64	18.87
	←	16.41	12.73

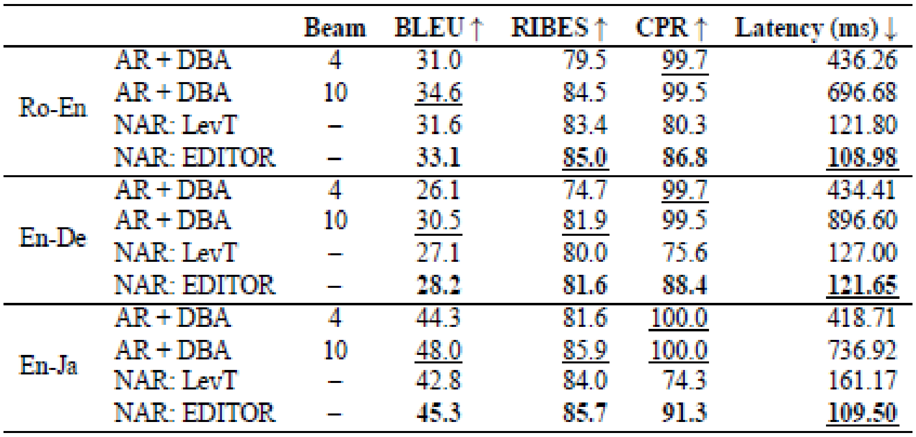
4.2.5 Other Experimental Results

Although it was not incorporated into the evaluation system, we assessed EDITOR's ability to incorporate lexical constraints into its outputs. This is motivated by the need to encourage MT systems to use terms that are consistent with the query when translating documents that have been found to be relevant. We evaluate on three translation tasks: Romanian-English, English- German, and English-Japanese translation with provided terminology constraints [41]. As shown in Table 18, we compare EDITOR with an autoregressive (AR) NMT model with con- strained beam search (AR+DBA) [148] and LevT [42], which is the SOTA non- auto-regressive (NAR) NMT model. For each metric, we underline the top scores among all models and boldface the top scores among NAR models based on the paired bootstrap test with $p < 0.05$ [149]. EDITOR decodes 6–7% faster than LevT on Ro-En and En-De, and 33% faster on En-Ja, while achieving comparable or higher BLEU and Rank-based Intuitive Bilingual Evaluation Score (RIBES). As in [42], we evaluate translation quality via case-sensitive tokenized BLEU and RIBES [150], which is more sensitive to word order differences. For lexically-constrained decoding, we report the constraint preservation rate (CPR) in the translation outputs. We quantify decoding speed using *latency* per sentence. This is computed as the average time[41] (in milliseconds) required to translate the test set using a batch size of one divided by the number of sentences in the test set.

Results show that EDITOR exploits soft lexical constraints more effectively than the LevT while also speeding up decoding as compared to the constrained beam search implementation used.

[41] Excluding the model loading time.

Table 18. Performance of EDITOR Compared with AR and NAR Translation Baselines on Lexically Constrained MT.



		Beam	**BLEU ↑**	**RIBES ↑**	**CPR ↑**	**Latency (ms) ↓**
Ro-En	AR + DBA	4	31.0	79.5	99.7	436.26
	AR + DBA	10	34.6	84.5	99.5	696.68
	NAR: LevT	–	31.6	83.4	80.3	121.80
	NAR: EDITOR	–	**33.1**	**85.0**	**86.8**	**108.98**
En-De	AR + DBA	4	26.1	74.7	99.7	434.41
	AR + DBA	10	30.5	81.9	99.5	896.60
	NAR: LevT	–	27.1	80.0	75.6	127.00
	NAR: EDITOR	–	**28.2**	**81.6**	**88.4**	**121.65**
En-Ja	AR + DBA	4	44.3	81.6	100.0	418.71
	AR + DBA	10	48.0	85.9	100.0	736.92
	NAR: LevT	–	42.8	84.0	74.3	161.17
	NAR: EDITOR	–	**45.3**	**85.7**	**91.3**	**109.50**

4.2.6 Technical Insights

Four main technical insights were generated as a result of this work, which reflect both contributions to the evaluation systems and additional research conducted in this area. Specifically, we find that:

1. Many of the tasks in the program were not truly low resource after collecting or crawling publicly available data. Translation quality is also improved through the creation of synthetic training data generated by back translation or pivoting.
2. Data pre-processing is key to building high quality systems. Data filtering and normalization, as well as dedicated handling of URLs and Twitter handles had a large impact on translation quality.
3. The needs of the end user (e.g., a CLIR system versus a human reader) dictate what makes an MT system useful, and this implies the need for distinct MT systems to meet specific needs. For example, SMT systems can help CLIR even when their BLEU scores are lower than NMT. But their ability to pass OOV words through can be distracting when presenting output to human users.
4. Tailoring MT to the upstream and downstream components (e.g., ASR, *n*-best outputs) proved useful. E2E training data would support investigations into the possible benefits of additional task integration techniques.

4.3 Data Collection and Language Identification

4.3.1 Text Data Collection

To support the other modules in SCRIPTS, we collected and shared a very large number of news articles scraped from a list of broadsheets, tabloids and local newspapers published online. The initial list of sources was extracted from the Wikipedia page “List of newspapers in the Philippines”[42] and we subsequently collected data for Swahili and Somali. The list of news websites and their URLs is shown in Appendix A.

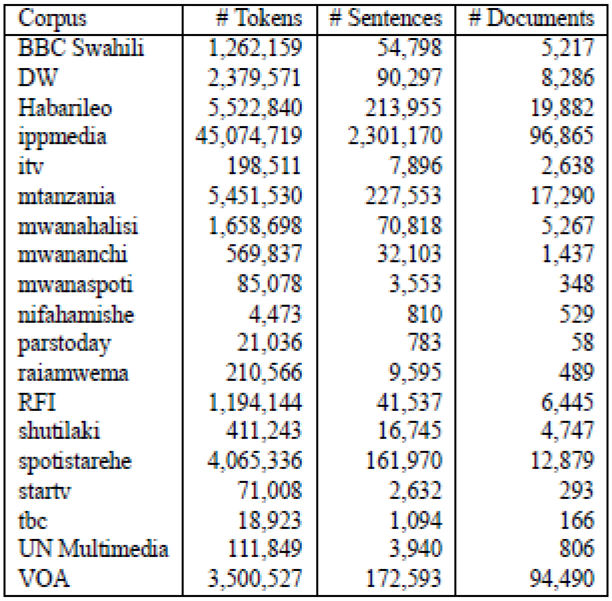
[42] https://en.wikipedia.org/wiki/List_of_newspapers_in_the_Philippines

For each news website, we scraped text, audio and subtitles if present. Text was segmented into sentences, normalized, and labeled with language tags.

We also collected text data from the *NYT*[43] and Reddit Summarization[44] datasets along with Global Voices'[45] multilingual blogs and articles. In total, we collected 30.5K Swahili posts and 2.5K for Tagalog.

Using our Speech Laboratory's Babler system [51], we also constructed a large corpus of blog posts scraped from blogger.com written in Swahili and Tagalog. We did this by seeding Babler with Swahili and Tagalog words extracted from the Leipzig corpus [151], and identifying the top scoring terms using a version of TF-IDF run on Microsoft BING web documents. The collection was also normalized and labeled with language tags.

Table 19. News Text Data for Swahili



Corpus	# Tokens	# Sentences	# Documents
BBC Swahili	1,262,159	54,798	5,217
DW	2,379,571	90,297	8,286
Habarileo	5,522,840	213,955	19,882
ippmedia	45,074,719	2,301,170	96,865
itv	198,511	7,896	2,638
mtanzania	5,451,530	227,553	17,290
mwanahalisi	1,658,698	70,818	5,267
mwananchi	569,837	32,103	1,437
mwanaspoti	85,078	3,553	348
nifahamishe	4,473	810	529
parstoday	21,036	783	58
raiamwema	210,566	9,595	489
RFI	1,194,144	41,537	6,445
shutilaki	411,243	16,745	4,747
spotistarehe	4,065,336	161,970	12,879
startv	71,008	2,632	293
tbc	18,923	1,094	166
UN Multimedia	111,849	3,940	806
VOA	3,500,527	172,593	94,490

For Tagalog, we obtained a total of 2,940,869 sentences of which 2,296,582 were tagged as Tagalog. For Swahili, we found 896,103 sentences, of which 828,534 were tagged as Swahili, as shown in Table 21.

We obtained additional text data from bilingual Tagalog/English and Swahili/English dictionaries. Tagalog was scraped from Glosbe,[46] including POS tags and morphology information where available. This produced a total of 26,864 Tagalog entries. The Swahili/ English diction-nary was scraped from,[47] with POS tags and morphology information also scraped when avail-able, for a total of 14,681 Swahili entries. We also found relevant Wiktionary Bilingual

[43] https://github.com/outerproduct/nyt-sum

[44] https://github.com/webis-de/webis-tldr-17-corpus

[45] https://globalvoices.org

[46] https://glosbe.com/tl/en

[47] http://africanlanguages.com/swahili/

Dictionaries for both languages and extracted data for each in two formats. The first type of dictionary was extracted from the translation pages of Wiktionary (e.g., en.wiktionary.org, tl.wiktionary.org and sw.wiktionary.org). These contained 8,132 Swahili entries and 10,886 Tagalog entries. The second type of dictionary was extracted using triangulation across the two target languages and a third one using wikt2dict.[48] These dictionaries generated 2,381 entries for Swahili and 2,275 for Tagalog.

Table 20. News Text Data for Tagalog.

Corpus	# Tokens	# Sentences	# Documents
abante-tonite	4,599,322	277,633	26,790
balita	15,668,728	794,073	125,223
bandera	23,686,772	1,442,994	84,558
gmanetwork	3,030,967	119,261	51,742
hataw	42,821	2,826	148
philstar	19,853,063	1,180,321	48,438
pinoyparazzi	12,266,823	902,174	57,987
pinoyweekly	1,936,271	107,200	5,949
remate	59,767,005	221,510	293,347

Table 21. Bing-scraped Text Data.

Language	Swahili	Tagalog
# Documents	91,305	51,630
# Documents in-target	90,171	49,175
# Sentences	10,948,666	13,735,047
# Sentences in-target	8,967,646	8,586,904

We also tagged Swahili and Tagalog CommonCrawl corpora with language tags, as detailed in Table 22.

Table 22. Number of Documents and Sentences in the CommonCrawl Collection.

Language	# Documents	# Sentences
Swahili	2.8 M	127.5 M
Tagalog	8.6 M	270 M

Finally, we used a 100-language corpus of parallel Biblical text[49] aligned at the verse level to collect 62,195 verses from complete Bibles available in Tagalog and 15,699 verses from the New Testament in Swahili.

In subsequent work in the BP, we searched YouTube for additional Bibles in the languages of interest, using the alignment from the text Bibles on the videos. We also worked to build a more robust YouTube scraper in order to identify more Tagalog and Swahili audio with captions.

To validate our collection processes, we ran a test of our web scraper's "massive" version, which

[48] https://github.com/juditacs/wikt2dict

[49] http://christos-c.com/bible/

scrapes all available websites for specified languages. We tested this massive scraper on the 200 top scored keywords for Tagalog and Swahili and requested from BING a maximum of 500 URLs, expecting to obtain at most 10,000 documents per language. Our results are as follows:

- Tagalog:
    - Total number of URLs found = 93037
    - Total number of docs found in-language (tgl) = 48099 (51.70%)
    - Total number of docs found in-language that were stored in this run = 23731(25.51%)
    - Total number of docs found OOL (eng) = 30193 (32.45%)
- Swahili:
    - Total number of urls found = 90270
    - Total number of docs found in-language (swa) = 63300 (70.12%)
    - Total number of docs found in-language that were stored in this run = 34928(38.69%)
    - Total number of docs found OOL (eng) = 14907 (16.51%)

Results using Language ID on these scraped documents were reasonably good: for 183,307 documents collected, 111,399 (61%) were in the language sought and 58,659 (32%) were stored in the procedure.

4.3.2 Audio Data Collection

To support ASR, we also scraped large amounts of LRL audio data. We obtained numerous audio clips from our work scraping news sources as described in Section 4.3.1. We did encounter certain challenges in this process: First, the videos we collected did not always represent news-type text as they often included advertisements. News videos were also never tagged by language. Finally, a majority of the captions we collected were automatically generated by You- Tube, either by ASR (often for the wrong language) or MT. To address some of these issues, we also collected audio clips from the same sources we used for our text news collection.

For each YouTube webpage described above, we scraped text, audio and subtitles if present. For audio, the following information was logged:

- Video identifier as assigned by the host website.
- Extension of the audio file.
- Language ID as assigned by the host website.
- Title of the video
- Language ID for the title assigned by CLD2.
- URL of the website (news source) where the video was found.
- URL of the website where the video is hosted.

The audio files were then down-sampled to 16khz to save disk space. The subtitles are in the video text tracks (*vtt)* format, which includes supplementary information such as subtitles, captions, descriptions, chapter, and metadata. These were automatically generated. Audio data collected in this manner for Swahili is shown in Table 23 and for Tagalog in Table 24.

Table 23. Audio Clips from Swahili News Websites with Multimedia Data.

Corpus	# WAV Files	Subtitles	Duration (h:m:s)
BBC Swahili	190	NO	23:39:35.46
ippmedia	1214	YES	98:46:16.45
itv	11	YES	00:49:03.96
mtanzania	10	YES	00:47:17.72
mwanahalisi	144	YES	14:30:27.19
mwananchi	1	YES	00:04:18.72
mwanaspoti	19	YES	01:13:02.96
nifahamishe	2	NO	02:01:07.64
RFI	21	YES	01:51:13.54
shutilaki	234	NO	20:07:36.05
spotistarehe	424	YES	52:52:09.92
startv	4	YES	01:36:40.35
UN Multimedia	28	YES	01:52:42.03
VOA	4087	NO	1299:37:56.22

We also employed Jenga[50] to scrape YouTube videos in LRLs. Jenga was developed in our laboratory which used user-submitted subtitles to collect videos in a requested language. We used the Swahili and Tagalog keyword lists provided by the IARPA Babel program to search YouTube for user-submitted subtitles for up to 20 results per keyword. Results are shown in Table 25.

We also scraped audio versions of the Bible from bible.com. For Tagalog, we obtained 99.5 hours of audio divided into 1,190 chapters. For Swahili, we downloaded 25.28 hours of audio from 260 chapters.

Table 24. Audio Clips Obtained from Tagalog News Websites with Multimedia Data.

Corpus	# WAV Files	Subtitles	Duration
abante-tonite	16	NO	00:15:06.91
balita	6	NO	00:15:07.74
bandera	36	NO	01:53:45.83
gmanetwork	3688	YES	203:00:27.24
philstar	45	YES	08:23:54.45
pinoyparazzi	310	YES	18:52:16.65
pinoyweekly	271	YES	26:39:32.61

[50] Referenced tool has since been removed from GitHub.com

Table 25. Metrics for the YouTube Corpus.

Metric	Swahili	Tagalog
# Keywords	4,455	3,805
# Videos Found	71,442	66,158
# User-transcribed Videos	1,458	546
# Utts	520,397	150,354
Utts Total Duration (h)	580.54	221.20
Utts Avg. Duration (s)	4.02	5.30

4.3.3 Language ID

We performed Language ID for our text corpora using the majority vote between three ad-hoc language taggers: CLD2 [152], Textcat [153], and Lingpipe [154]. Majority vote was preferred over a single classifier's prediction because aggregate predictions work better for LRLs [51]. When majority vote ties to 1-1-1, we fell back to the single method with highest confidence score. Documents were tagged with language identifiers at both the document level and at the sentence level.

To identify code-switched sentences, we developed a method that we termed *anchoring*. An *anchor* is a word that belongs to only one language among a large pool of languages and are especially useful for detecting code-switching (switching between languages) in documents or sentences. For example, if a sentence contains anchors from two different languages, it can be concluded with a certain confidence that the sentence is code-switched. Similarly, if the sentence contains an anchor from language L_1 but its language identifier is L_2, it can also be concluded that the sentence is code-switched in L_1+L_2. For MATERIAL, we tagged word-level anchor annotations on all the scraped data in order to study the impact of code-switched data on the different NLP modules of our system pipeline. We computed weak anchors using the Leipzig corpus, which included a total of 2,758 Swahili words and 21,849 Tagalog words.

4.3.4 Technical Insights

Overall, our news and YouTube collected data supported major improvements in SCRIPTS program, particularly in ASR where it improved audio-based performance over the initial ASR error rate of 50%. Specifically, performance improved by 19% for Swahili and 12% for Somali. Text data we collected for ASR LM also was quite useful: in Swahili, for example, there were relative WER gains of 5% on NB and 28% on WB data types. LM for MT was also improved thanks to the significantly higher volume of training data we provided in the evaluation languages. The bilingual dictionaries we collected were also used for intrinsic evaluation of multilingual word embeddings.

We also submitted several systems that made use of language ID-based text and speech filtering. The first one was for Swahili, where the baseline system with document filtering obtained 0.299 AQWV score. The system that filtered out documents using both speech and text language identification, meanwhile, achieved a 0.302 AQWV score - equivalent to a 1% relative improvment. For Somali, the baseline system achieved a 0.187 AQWV score, whereas the submission system submission that used document filtering with text language ID achieved 0.190 AQWV - a 1.60% relative improvement - and one using speech language ID filtering achieved 0.191 AQWV - a 2.14% relative improvement. Finally, a system using both forms of filtering achieved 0.193 AQWV, for a total 3.21% relative improvement.

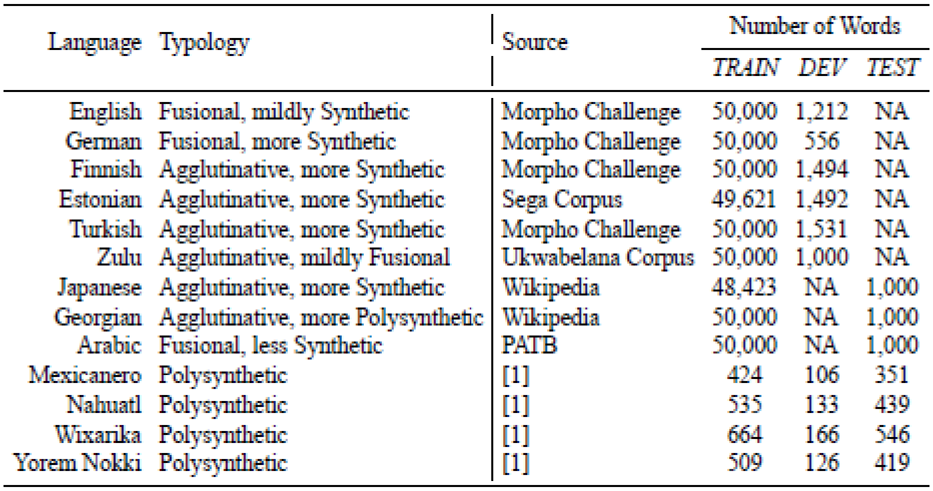
4.4 Text Processing

In this section, we present the experimental settings (e.g., the languages, data, metrics) and the results and findings for morphological segmentation (MorphAGram) and unsupervised POS tagging portions of this research.

4.4.1 MorphAGram - Languages, Data, Evaluation Settings

We considered 13 languages that were spread across the typology spectrum and for which morphologically segmented datasets were available for evaluation. Six out of the 13 were development languages that we used to derive the main conclusions concerning our grammar definitions, learning settings and the automatic tailoring of grammars for unseen languages. These languages are English, German, Finnish, Estonian, Turkish and Zulu. The remaining languages are the test ones, specifically: Japanese, Georgian, Arabic, Mexicanero, Nahuatl (Mexicano), Wixarika (Huichol) and Mayo (Yorem Nokki). Brief descriptions for the typologies of the languages, along with information about the datasets, are listed in Table 26. In addition, we provide the information about the data (unsegmented words) we used for training MorphA-Gram for MATERIAL languages (note that except for Georgian, we do not have gold segmented data for MATERIAL languages so we cannot make that evaluation).

Table 26. Typological and Data-related Information per Experimental Language for MorphAGram.



Language	Typology	Source	Number of Words		
			TRAIN	*DEV*	*TEST*
English	Fusional, mildly Synthetic	Morpho Challenge	50,000	1,212	NA
German	Fusional, more Synthetic	Morpho Challenge	50,000	556	NA
Finnish	Agglutinative, more Synthetic	Morpho Challenge	50,000	1,494	NA
Estonian	Agglutinative, more Synthetic	Sega Corpus	49,621	1,492	NA
Turkish	Agglutinative, more Synthetic	Morpho Challenge	50,000	1,531	NA
Zulu	Agglutinative, mildly Fusional	Ukwabelana Corpus	50,000	1,000	NA
Japanese	Agglutinative, more Synthetic	Wikipedia	48,423	NA	1,000
Georgian	Agglutinative, more Polysynthetic	Wikipedia	50,000	NA	1,000
Arabic	Fusional, less Synthetic	PATB	50,000	NA	1,000
Mexicanero	Polysynthetic	[1]	424	106	351
Nahuatl	Polysynthetic	[1]	535	133	439
Wixarika	Polysynthetic	[1]	664	166	546
Yorem Nokki	Polysynthetic	[1]	509	126	419

For the *Scholar-Seeded* affixes described in Section 3.4.1, we mainly relied on the Wiktionary to collect the prefixes and suffixes of the language of interest, supplemented with additional ones from grammar pages on the Web if necessary. In order to preserve the low resource setting, however, we restricted the process of collecting the list of affixes less than two hours in length per language. For all the languages, we trained our models using the training sets (*TRAIN*) *without* seeing gold standard segmentation. We conducted our experiments in a transductive learning scenario, where the unsegmented words in the evaluation set were included in the training set, which is common in the evaluation of unsupervised morphological segmentation [155], [55], [156]. Notably, we do not see significant gains in the performance when using an inductive approach in which the unsegmented words in the evaluation set are kept separate from the training set. For training, we ran the sampler for 500 optimization iterations for all the languages. Annealing was found to have no positive impact and so was not used; all the hyper-parameters of the model and the probabilities of the production rules are automatically inferred. Because the sampler is non-deterministic, we compute all the evaluation results as the average of five runs.

We evaluated the performance of *MorphAGram*, our morphological-segmentation framework, using the classical evaluation method of Boundary Precision and Recall (BPR). BPR measures the ability of the system to detect segmentation boundaries by comparing the boundaries in the proposed segmentation to those in the reference.

For MATERIAL languages, we used the *Scholar-Seeded* setting to train our morphological-segmentation models. We compiled the lists of seeded affixes from Wiktionary and trained our models using the 50,000 most frequent words seen in the BUILD and DEV data packages for each language. We chose the scholar-seeded learning setting as it shows best performance in general across a variety of language. We could not evaluate the performance for MATERIAL languages except Georgian as we did not have gold segmentation.

4.4.2 MorphAGram - Results

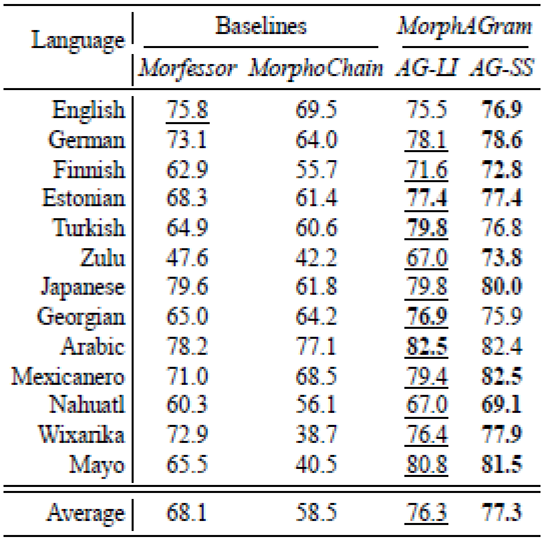
We compared the performance of *MorphAGram* with two strong baselines, Morfessor[51] and MorphoChain.[52] As shown in Table 27, our *AG Scholar-Seeded* setup (*AG-SS*), outperforms our fully unsupervised setup, AG Language Independent (*AG-LI)*, in nine individual languages and on average, achieving an average relative error reduction of 5.1%. Comparing *MorphAGram* to the baselines, the *AG-LI* configuration outperforms both *Morfessor* and *MorphoChain* when evaluated on all the languages except English, with average relative error reductions of 22.8% and 40.7%, respectively. In the case of English, *Morfessor* outperforms *AG-LI* by an absolute 0.3%, while it comes second to *AG-SS* by an absolute 1.1%. The best overall result per language is in **bold**. The best language-independent result per language is underlined.

A notable capability of MorphAGram is its ability to handle polysynthetic languages (in which a word may contain several morphemes) even in low resource setups of only about 1,000 available words. Table 28 illustrates the performance of *MorphAGram*'s unsupervised system versus four *supervised* neural systems by [1] - namely *S2S*, Conditional-Random Fields (*CRF)*, Best Multi-Task Training *(BestMTT)* and Best Data-Augmentation (*BestDA)*, by evaluating on *TEST* using the BPR metric, in terms of F1-score. MorphAGram outperforms all of the super- vised neural systems by [1] (including *BestMTT*, currently the best multi-task training system and BestDA, currently the best data augmentation system - when evaluated on Mayo using the same training and evaluation sets; we, however, do not use the gold segmentation for training. In the case of Nahuatl, AG-SS is only 0.5% behind the best supervised system, CRF. The performance gaps in Mexicanero and Wixarika are also relatively small, especially given the supervised nature of the baselines. The best result per language is in bold.

[51] https://morfessor.readthedocs.io/en/latest/

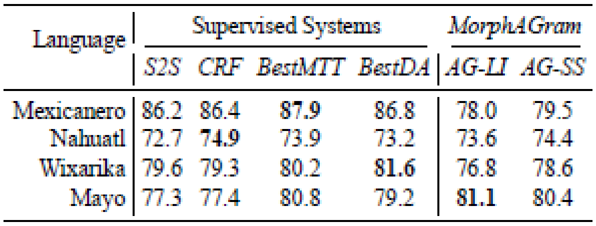
[52] https://github.com/karthikncode/MorphoChain/blob/production2/README.md

Table 27. MorphAGram vs. Morfessor and MorphoChain (BPR F1-score).



Language	Baselines		*MorphAGram*	
	Morfessor	*MorphoChain*	*AG-LI*	*AG-SS*
English	75.8	69.5	75.5	**76.9**
German	73.1	64.0	78.1	**78.6**
Finnish	62.9	55.7	71.6	**72.8**
Estonian	68.3	61.4	**77.4**	**77.4**
Turkish	64.9	60.6	**79.8**	76.8
Zulu	47.6	42.2	67.0	**73.8**
Japanese	79.6	61.8	79.8	**80.0**
Georgian	65.0	64.2	**76.9**	75.9
Arabic	78.2	77.1	**82.5**	82.4
Mexicanero	71.0	68.5	79.4	**82.5**
Nahuatl	60.3	56.1	67.0	**69.1**
Wixarika	72.9	38.7	76.4	**77.9**
Mayo	65.5	40.5	80.8	**81.5**
Average	68.1	58.5	76.3	**77.3**

Table 28. MorphAGram vs. Kann et al. 2018 [1] (BPR F1-score).

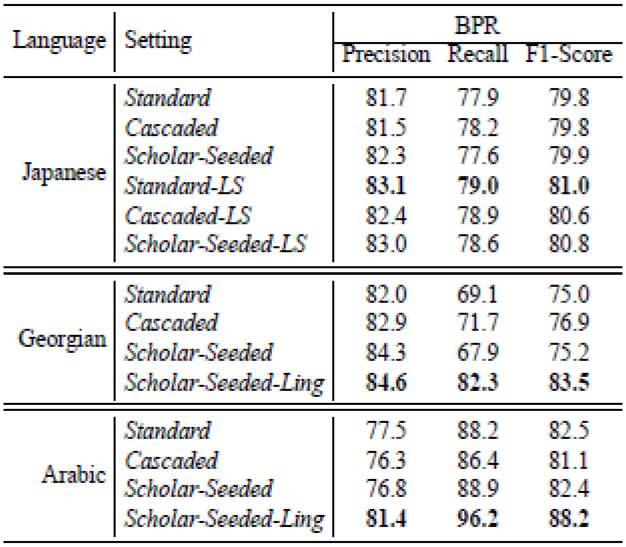


Language	Supervised Systems				*MorphAGram*	
	S2S	*CRF*	*BestMTT*	*BestDA*	*AG-LI*	*AG-SS*
Mexicanero	86.2	86.4	**87.9**	86.8	78.0	79.5
Nahuatl	72.7	**74.9**	73.9	73.2	73.6	74.4
Wixarika	79.6	79.3	80.2	**81.6**	76.8	78.6
Mayo	77.3	77.4	80.8	79.2	**81.1**	80.4

4.4.2.1 Incorporating Linguistic Priors

Table 29 reports the morphological segmentation performance with the incorporation of linguistic priors into the PrStSu+SM grammar in the form of grammar definition (LS) for Japanese, and linguist-provided (Ling.) affixes for Georgian and Arabic (see Section 3.4.1 for details). The use of linguistic priors consistently improves the performance in all the settings, and all the improvements are statistically significant for *p-value* < 0.01. The best result per is in **bold**.

Table 29. The Performance for Japanese, Georgian and Arabic with and without the Use of Linguistic Priors within the *PrStSu+SM* Grammar.

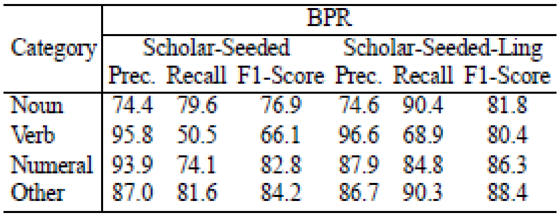


Language	Setting	BPR		
		Precision	Recall	F1-Score
Japanese	*Standard*	81.7	77.9	79.8
	Cascaded	81.5	78.2	79.8
	Scholar-Seeded	82.3	77.6	79.9
	Standard-LS	**83.1**	**79.0**	**81.0**
	Cascaded-LS	82.4	78.9	80.6
	Scholar-Seeded-LS	83.0	78.6	80.8
Georgian	*Standard*	82.0	69.1	75.0
	Cascaded	82.9	71.7	76.9
	Scholar-Seeded	84.3	67.9	75.2
	Scholar-Seeded-Ling	**84.6**	**82.3**	**83.5**
Arabic	*Standard*	77.5	88.2	82.5
	Cascaded	76.3	86.4	81.1
	Scholar-Seeded	76.8	88.9	82.4
	Scholar-Seeded-Ling	**81.4**	**96.2**	**88.2**

In the case of Japanese, the use of a language-specific grammar leads to the best performance in terms of precision, recall and F1-score, achieving relative error reductions of 6.0%, 4.2% and 4.5% in BPR F1-score in the *Standard*, *Cascaded* and *Scholar-Seeded* settings, respectively. In the case of Georgian, the use of linguist-provided affixes yields the best results, with relative error reductions of 33.2% in BPR F1-score over the regular, *Scholar-Seeded* setting, which relies on affixes of lower quality. A similar pattern is seen for Arabic, where the use of linguist-pro-vided affixes yields the best performance in terms of precision, recall and F1-score, achieving an error reduction of 32.9% in BPR F1-score over the *Scholar-Seeded* setting. In both Georgian and Arabic, the use of linguist-provided affixes impacts recall more than precision, as the sampler knows about the most common affixes in the underlying language; these represent the majority of the affixes seen in the gold segmentation. However, precision also improves as the probability of expanding to existing production rules that represent the seeded affixes is higher than the probability of expanding new subtrees representing unseen affixes.

As shown in Table 30, we also conducted an analysis per POS for Georgian, as our linguist provided (Ling.) annotation of word categories. While we see improvements across the board, we notice the highest improvements for verbs, because the linguist provided more affixes relevant to verbal constructions.

Table 30. Category-wise Morphological-Segmentation Performance for Georgian Using the BPR.



Category	BPR					
	Scholar-Seeded			Scholar-Seeded-Ling		
	Prec.	Recall	F1-Score	Prec.	Recall	F1-Score
Noun	74.4	79.6	76.9	74.6	90.4	81.8
Verb	95.8	50.5	66.1	96.6	68.9	80.4
Numeral	93.9	74.1	82.8	87.9	84.8	86.3
Other	87.0	81.6	84.2	86.7	90.3	88.4

4.4.3 MorphAGram - Technical Insights

The ability of AGs to include linguistic priors (either as grammar design or linguist-provided affixes) improves performance. MorphAGram - which is based on AGs - achieves the SOTA for unsupervised morphological segmentation results across a range of languages of diverse morphological complexity.

4.4.4 POS Tagging - Languages, Data, Evaluations Settings

We provide below description of our experimental settings for POS tagging including languages, data and evaluation metrics.

4.4.4.1 Word-based Alignments Experiments

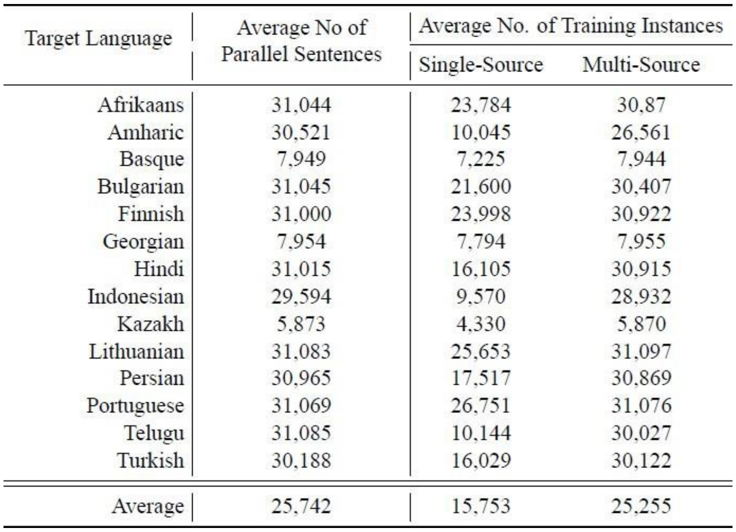
We conducted our cross-lingual POS-tagging experiments on six high resource languages and 14 simulated low resource target ones of diverse morphological typologies. Of these 14 target languages, five are MATERIAL languages for which we have POS tagged data for evaluation, generating a total of 84 target-source language pairs. We choose widely spoken high resource languages, because parallel texts for LRL are highly likely to include at least one of the high resource languages, especially when translating religious books, movie scripts and user manuals. These selected high resource languages are English (Indo-European (IE), Germanic), Spanish (IE, Romance), French (IE, Romance), German (IE, Germanic), Russian (IE, Slavic) and Arabic (Afro-Asiatic, Semitic). In the case of target languages that are in fact high resource, we simulate a low resource scenario where the POS tagging is performed in a fully unsupervised fashion. The target languages (**MATERIAL languages appear in bold**) are: Afrikaans (IE, Germanic), Amharic (Afro-Asiatic, Semitic), Basque (language isolate), **Bulgarian** (IE, Slavic), Finnish (Uralic, Finnic), **Georgian** (Kartvelian), Hindi (IE, Hindi), Indonesian (Austronesian, Malayo-Sumbawan), **Kazakh** (Turkic, Northwestern), **Lithuanian** (IE, Baltic), **Farsi** (IE, Iranian), Portuguese (IE, Romance), Telugu (Dravidian, South Central) and Turkish (Turkic, South-western). We use the multilingual parallel Bible corpus[53] by [157] as the source of our parallel data for all languages apart from Georgian and Kazakh, for which we collected the biblical texts from the MissingBibleVerses corpus.[54] The full text of the Bible is available for all source and target languages except for Basque, Georgian and Kazakh, where only the New Testament is available. However, the small volume of data available in Basque and the fact that it is a language isolate make it an ideal case study of cross-lingual learning in a low resource scenario.

The second column of Table 31 lists the average number of parallel sentences per target language across the source languages. The third column, meanwhile, contains the corresponding average number of training sentences after applying the sentence selection mechanism de- scribed in Section 3.4.2 for single source projection; the fourth column contains this average for multi-source projection. For single-source projection, Indonesian, Telugu and Amharic experience the maximum loss in the number of sentences selected as training instances, with relative reductions of 67.7%, 67.4% and 67.1%, respectively; the average relative reduction across the target languages is 38.8%. It is noteworthy to mention that we run the approach on verses as opposed to sentences, which are not equivalent in the rare cases where a verse contains multiple sentences or a sentence spans multiple verses.

[53] http://christos-c.com/Bible

[54] https://github.com/cysouw/MissingBibleVerses

Table 31. Average Number of Alignment and Training Sentences per Target Language Using the Bible Parallel Data Source (Single and Multi-Source Projection).



Target Language	Average No of Parallel Sentences	Average No. of Training Instances	
		Single-Source	Multi-Source
Afrikaans	31,044	23,784	30,87
Amharic	30,521	10,045	26,561
Basque	7,949	7,225	7,944
Bulgarian	31,045	21,600	30,407
Finnish	31,000	23,998	30,922
Georgian	7,954	7,794	7,955
Hindi	31,015	16,105	30,915
Indonesian	29,594	9,570	28,932
Kazakh	5,873	4,330	5,870
Lithuanian	31,083	25,653	31,097
Persian	30,965	17,517	30,869
Portuguese	31,069	26,751	31,076
Telugu	31,085	10,144	30,027
Turkish	30,188	16,029	30,122
Average	25,742	15,753	25,255

We also noticed that doing multi-source projections increases the number of training instances for all languages. For testing, we used the test datasets of the UD project, UD-v2.5 [158] to evaluate our tagging models in terms of POS accuracy. The corpora are *Afrikaans-AfriBooms*, *Amharic-ATT*, *Basque-BDT*, *Bulgarian-BTB*, *Finnish-TDT*, *Hindi-HDTB*, *Indonesian-GSD*, *Kazakh-KTB*, *Lithuanian-ALKSNIS*, *Persian-Seraji*, *Portuguese- Bosque*, *Telugu-MTG* and *Turkish-IMST*. We also report our results on older versions of the UD project in order to compare to the SOTA systems, when needed. One exception is Georgian, where it is not part of UD. Therefore, we developed a small POS-tagged dataset of 100 sentences for Georgian,[55] following the UD- tagging schema. The sentences are taken from the Modern Georgian and Political texts sub- corpora of the Georgian National Corpus,[56] and are hand tagged and carefully reviewed by a linguist who specializes in and speaks Georgian as a second language. Finally, we evaluate our approach for cross-lingual POS tagging via alignment and projection versus zero-shot model transfer on Japanese as a case study, where we use the Japanese test set from the 2017 Conference on Computational Natural Language Learning (CoNLL-2017) shared task [158] for evaluation.

4.4.4.2 Stem-based Alignment Experiments

We selected eight morphologically complex target languages on which to evaluate our word-based approach: six that are largely agglutinative (Basque, Finnish, Georgian, Kazakh, Telugu, and Turkish), Amharic (where many morphological alterations rely on consonantal roots), and the less morphologically rich Indonesian. However, we use the same set of source languages, for a total of 48 language pairs.

We also experimented with multi-source setups. Finally, we chose to use the New Testament instead of the entire Bible as the source of parallel data for alignment and projection in the stem-

[55] https://github.com/rnd2110/unsupervised-cross-lingual-POS-tagging/blob/main/data/KAT-eval.txt
[56] http://gnc.gov.ge

based approach for two reasons: first, three target languages only have the New Testament (Georgian, Basque and Kazakh); second, we wanted to demonstrate the efficiency of stem-based alignment and projection, where the use of the stem compensates for the lack of adequate amount of data. We however use the same evaluation datasets as before.

4.4.5 POS Tagging (Neural Model) - Results

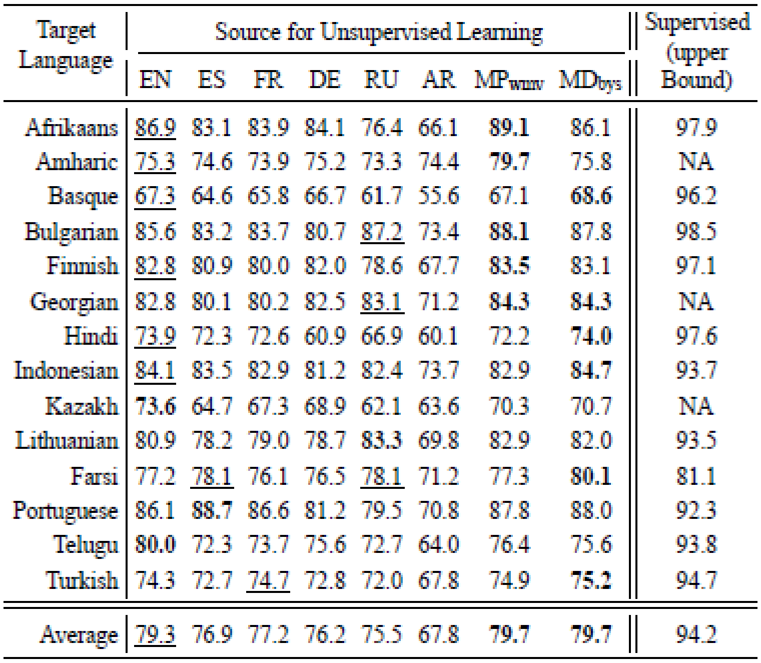
We have conducted most of our experimentation with the Neural POS tagger, as that was superior to the Average Perceptron Model in our earlier experiments. Here we present the main results of the work; more results can be found in [159].

4.4.5.1 Word-based Alignment Experiments

Table 32 reports the accuracy of our POS taggers for all the 84 language pairs, the average performance per source and target language, as well as the multi-source projection and multi-source decoding. For the latter two, we reported the base settings, which are maximum voting for the multi-source projection and Bayesian Inference for multi-source decoding. The last column reports the upper bound supervised performance using Stanford's Stanza (https://stanfordnlp.github.io/stanza/pos.html). The supervised performance is unavailable for Amharic and Georgian due to the unavailability of UD training data and for Kazakh. The best results per target language and per source language on average across the target languages is in bold. The last column reports the upper bound supervised performance using Stanza.

Table 32. The POS-tagging Performance (Accuracy) when Using the Bible as the Source of Parallel Data.

MPwmv stands for multi-source projection using weighted maximum voting; MDbys stands for multi-source decoding using Bayesian inference.



Target Language	Source for Unsupervised Learning								Supervised (upper Bound)
	EN	ES	FR	DE	RU	AR	MP$_{wmv}$	MD$_{bys}$	
Afrikaans	86.9	83.1	83.9	84.1	76.4	66.1	**89.1**	86.1	97.9
Amharic	75.3	74.6	73.9	75.2	73.3	74.4	**79.7**	75.8	NA
Basque	67.3	64.6	65.8	66.7	61.7	55.6	67.1	**68.6**	96.2
Bulgarian	85.6	83.2	83.7	80.7	87.2	73.4	**88.1**	87.8	98.5
Finnish	82.8	80.9	80.0	82.0	78.6	67.7	**83.5**	83.1	97.1
Georgian	82.8	80.1	80.2	82.5	83.1	71.2	**84.3**	**84.3**	NA
Hindi	73.9	72.3	72.6	60.9	66.9	60.1	72.2	**74.0**	97.6
Indonesian	84.1	83.5	82.9	81.2	82.4	73.7	82.9	**84.7**	93.7
Kazakh	**73.6**	64.7	67.3	68.9	62.1	63.6	70.3	70.7	NA
Lithuanian	80.9	78.2	79.0	78.7	**83.3**	69.8	82.9	82.0	93.5
Farsi	77.2	78.1	76.1	76.5	78.1	71.2	77.3	**80.1**	81.1
Portuguese	86.1	**88.7**	86.6	81.2	79.5	70.8	87.8	88.0	92.3
Telugu	**80.0**	72.3	73.7	75.6	72.7	64.0	76.4	75.6	93.8
Turkish	74.3	72.7	74.7	72.8	72.0	67.8	74.9	**75.2**	94.7
Average	79.3	76.9	77.2	76.2	75.5	67.8	**79.7**	**79.7**	94.2

The overall approach achieves an average POS accuracy of 75.5% across all the language pairs. However, there is a noticeable variance in the performance of the different taggers. Specifically, languages that belong to the same family transfer best across each other. For instance, English and

German yield the best results for Afrikaans (IE, Germanic), while Spanish and Portuguese are the best performing language pair (IE, Romance), and Russian is the best source for Bulgarian (IE, Slavic). One exception is the case of transferring from Arabic to Amharic (Afro-Asiatic, Semitic). One possible reason is that the Arabic analyzer does not follow the UD guidelines.

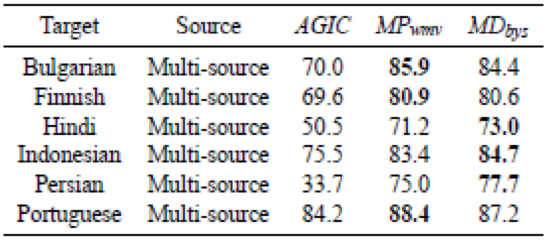
Arabic also exhibits a high degree of morphological complexity, which affects the performance of *all* the taggers that use Arabic as the source.

Since English is the highest resourced language, transferring from English yields the best performance for the eight target languages where its morphological annotation guidelines were the basis for those languages, namely Afrikaans, Amharic, Basque, Finnish, Hindi, Indonesian, Kazakh and Telugu. English also gives the best performance on average, with an average relative error reduction of 9.2% over French, the second best on average performing source language. However, while French only yields the best performance for Turkish, Russian is the best source language for four target languages: Bulgarian, Georgian, Lithuanian and Persian. Arabic is the lowest performing source language because its morphological complexity involves inflection, fusion and affixation, and because its analyzer does not follow the UD guidelines. The performance of the target languages is mainly impacted by three factors: morphological complexity, source language similarity and data availability.

We found that using multi-source languages via multi-source projection and multi-source decoding improved the performance for all target languages except Kazakh, Lithuanian, Portuguese and Telugu. The improvement through multi-source projection is due to the significant decrease in the percentage of OOVs and the significant increase in the number of training instances, along with the improved quality of the projected tags. On the other hand, the improvement by multi- source decoding is due to combining the outputs of different models that each can perform best on different sets of tags, where the models are based on different training sets learned through different source languages. For cases where the multi-source setups improved over the best single source performance, the results are statistically significant for *p-value* < 0.01, except for Hindi.

In Table 33, we also show that multi-source approaches outperform the SOTA unsupervised and semi-supervised cross-lingual POS taggers that rely on learning from a large number of source languages. We first compare our best setup of multi-source projection, MP_{wmv}, and our best setup of multi-source decoding, MD_{bys}, to the unsupervised multi-source system by [82], denoted by Application Gateway Ingress Controller (*AGIC)*. We evaluate the performance of our system versus *AGIC* on the shared target languages, namely Bulgarian, Finnish, Hindi, Indonesian, Persian and Portuguese. Despite the use of fewer source languages and a less suitable source of parallel data, our approach outperforms *AGIC* on all the target languages, with average relative error reductions of 51.5% and 52.0% in the MP_{wmv} and MD_{bys} setups, respectively.

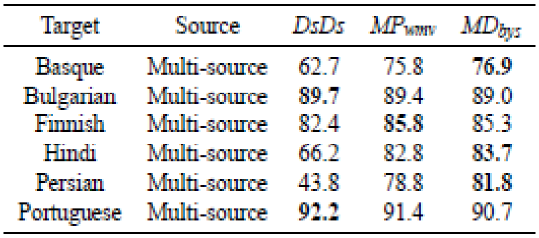
Table 33. Comparison to SOTA Unsupervised System AGIC (POS Accuracy).



Target	Source	*AGIC*	MP_{wmv}	MD_{bys}
Bulgarian	Multi-source	70.0	**85.9**	84.4
Finnish	Multi-source	69.6	**80.9**	80.6
Hindi	Multi-source	50.5	71.2	**73.0**
Indonesian	Multi-source	75.5	83.4	**84.7**
Persian	Multi-source	33.7	75.0	**77.7**
Portuguese	Multi-source	84.2	**88.4**	87.2

Next, we compare our approach to the SOTA multi-source semi-supervised system by [83], denoted by Distant Supervision from Disparate Sources (*DsDs)*. We evaluate our system versus *DsDs* on the shared target languages, namely Basque, Bulgarian, Finnish, Hindi, Persian, and Portuguese, using the development sets of UD-v2.1 (except for Basque, where, as in [83], the test set is used instead). Despite the use of fewer source languages, a less suitable source of parallel data and a fully unsupervised approach that does not make use of external language-dependent resources, our approach outperforms *DsDs* on all the target languages except Bulgarian and Portuguese, with average relative error reductions of 25.4% and 24.8% in the MP_{wmv} and MD_{bys} setups, respectively (Table 34).

Table 34. Comparison to SOTA Semi-Supervised System DsDs (POS Accuracy).



Target	Source	*DsDs*	MP_{wmv}	MD_{bys}
Basque	Multi-source	62.7	75.8	**76.9**
Bulgarian	Multi-source	**89.7**	89.4	89.0
Finnish	Multi-source	82.4	**85.8**	85.3
Hindi	Multi-source	66.2	82.8	**83.7**
Persian	Multi-source	43.8	78.8	**81.8**
Portuguese	Multi-source	**92.2**	91.4	90.7

4.4.5.2 Stem-based Alignment Experiments

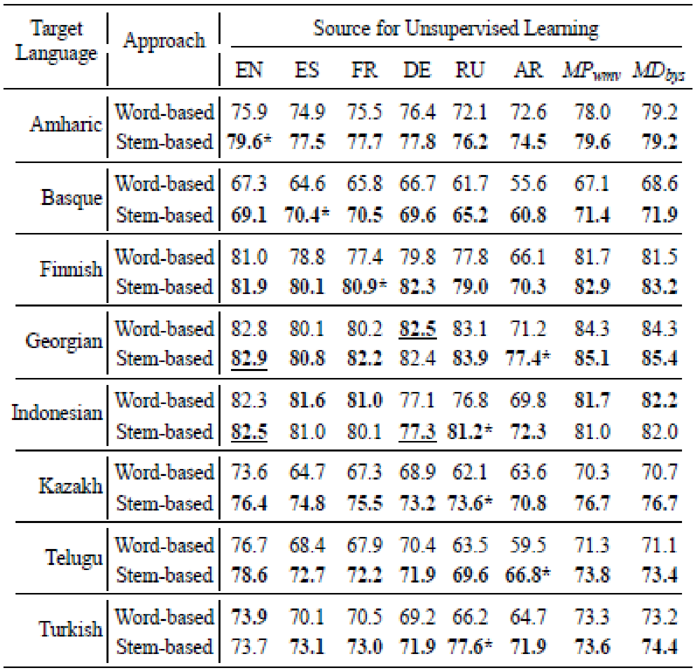
Table 35 reports the accuracy of our POS taggers in the stem-based setups compared to the word-based setups in both single-source and multi- source configurations. The single-source stem-based approach outperforms the single-source word-based one in 44 out of 48 language pairs; the five language pairs that benefit more from word-based alignment and projection are {Georgian, German}, {Indonesian, Spanish}, {Indonesian, French} and {Turkish, English}. For multi-source setups, the stem-based approach always improves performance, except in the case of Indonesian, where the two methods perform quite similarly. The best result per target-source language pair is in **bold**. The highest relative error reduction in the stem-based approach per target language is marked by *. The improvements in the stem-based setups that are not statistically significant for *p-value* < 0.01 are underlined.

In addition to the results presented here based on the Bible data, we conducted an experiment for Georgian where we added the MATERIAL Build data to the available New Testament Bible data when the source is English. The accuracy is 83.7, which is better than the single- source English model trained only on Bible data for word-based models (82.8), but worse than using multi- source projections or decoding using just Bible data (84.3) and worse than stem-based multi- source (85.4).

4.4.6 POS Tagging - Technical Insights

There are three key takeaways for our work on unsupervised cross-lingual POS tagging. First, building reliable training data via cross-lingual projection requires the use of bidirectional alignments, type and token constraints, alignment quality *and* density of projected tags. Second, using stem as the unit of abstraction instead of a word for cross-lingual projection is helpful, especially for morphologically complex languages. Third, learning from multiple source languages (either via projection or decoding) generates significant performance improvements across languages.

Table 35. The POS-Tagging Accuracy of the Word-Based and Stem-Based Setups When Using the New Testament as the Source of Parallel Data.



Target Language	Approach	Source for Unsupervised Learning							
		EN	ES	FR	DE	RU	AR	MP_{wmv}	MD_{bys}
Amharic	Word-based	75.9	74.9	75.5	76.4	72.1	72.6	78.0	79.2
	Stem-based	**79.6***	**77.5**	**77.7**	**77.8**	**76.2**	**74.5**	**79.6**	**79.2**
Basque	Word-based	67.3	64.6	65.8	66.7	61.7	55.6	67.1	68.6
	Stem-based	**69.1**	**70.4***	**70.5**	**69.6**	**65.2**	**60.8**	**71.4**	**71.9**
Finnish	Word-based	81.0	78.8	77.4	79.8	77.8	66.1	81.7	81.5
	Stem-based	**81.9**	**80.1**	**80.9***	**82.3**	**79.0**	**70.3**	**82.9**	**83.2**
Georgian	Word-based	82.8	80.1	80.2	**82.5**	83.1	71.2	84.3	84.3
	Stem-based	82.9	**80.8**	**82.2**	82.4	**83.9**	**77.4***	**85.1**	**85.4**
Indonesian	Word-based	82.3	**81.6**	**81.0**	77.1	76.8	69.8	**81.7**	**82.2**
	Stem-based	82.5	81.0	80.1	77.3	**81.2***	**72.3**	81.0	82.0
Kazakh	Word-based	73.6	64.7	67.3	68.9	62.1	63.6	70.3	70.7
	Stem-based	**76.4**	**74.8**	**75.5**	**73.2**	**73.6***	**70.8**	**76.7**	**76.7**
Telugu	Word-based	76.7	68.4	67.9	70.4	63.5	59.5	71.3	71.1
	Stem-based	**78.6**	**72.7**	**72.2**	**71.9**	**69.6**	**66.8***	**73.8**	**73.4**
Turkish	Word-based	**73.9**	70.1	70.5	69.2	66.2	64.7	73.3	73.2
	Stem-based	73.7	**73.1**	**73.0**	**71.9**	**77.6***	**71.9**	**73.6**	**74.4**

4.5 CLIR

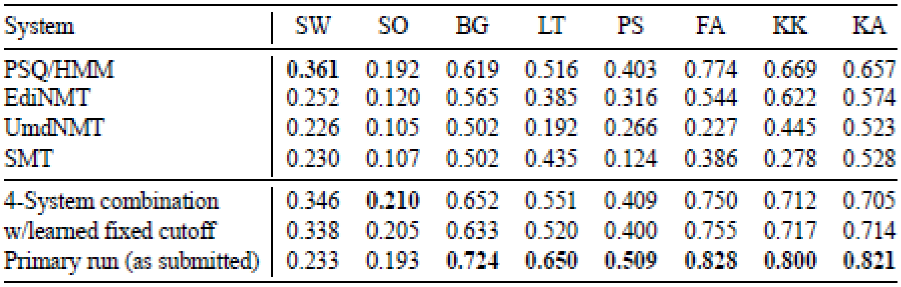
4.5.1 Results and Findings

Tables 36 and 37 show representative results on the text and speech EVAL collections for eight of the program's nine languages.[57] The top five rows of data in each table show results using MQWV, while the bottom two rows show results using AQWV. The results under comparable conditions are shown in the first six rows of each table, with those submitted during the scheduled evaluation period for each language as our primary run in the last row of each table. Starting with Bulgarian: BG in OP1, our primary submitted run outperformed the four-system combination that we have reported here for comparative purposes, reflecting language-specific choices of system configurations that were used in those evaluations. On line five of each table, an increasing trend is evident in the four-system combination MQWV reflecting the increasing maturity of the ASR, MT, and ranking system components, though language and collection differences make this trend more suggestive than confirmatory. A similar pattern is seen in the primary runs. Notable outliers are Somali (SO) and Pashto (PT), which generally yielded somewhat lower component and four-system combination MQWV as well as lower submitted primary

[57] The EVAL collection is not available for post-hoc analysis on Tagalog

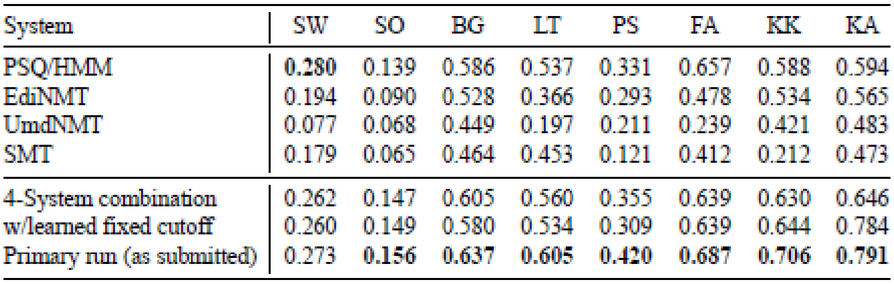
run AQWV than did other languages. This suggests somewhat greater difficulty in modeling those two languages, although again collection and differences are a confounding variable.

Table 36. Comparing EVAL Text Systems.



System	SW	SO	BG	LT	PS	FA	KK	KA
PSQ/HMM	**0.361**	0.192	0.619	0.516	0.403	0.774	0.669	0.657
EdiNMT	0.252	0.120	0.565	0.385	0.316	0.544	0.622	0.574
UmdNMT	0.226	0.105	0.502	0.192	0.266	0.227	0.445	0.523
SMT	0.230	0.107	0.502	0.435	0.124	0.386	0.278	0.528
4-System combination	0.346	**0.210**	0.652	0.551	0.409	0.750	0.712	0.705
w/learned fixed cutoff	0.338	0.205	0.633	0.520	0.400	0.755	0.717	0.714
Primary run (as submitted)	0.233	0.193	**0.724**	**0.650**	**0.509**	**0.828**	**0.800**	**0.821**

Table 37. Comparing EVAL Speech Systems.

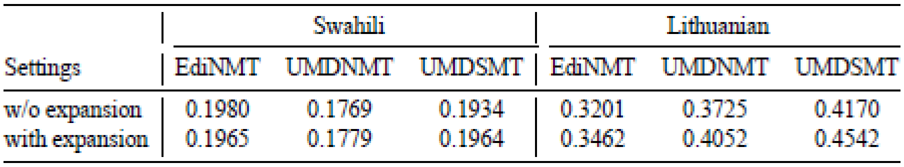


System	SW	SO	BG	LT	PS	FA	KK	KA
PSQ/HMM	**0.280**	0.139	0.586	0.537	0.331	0.657	0.588	0.594
EdiNMT	0.194	0.090	0.528	0.366	0.293	0.478	0.534	0.565
UmdNMT	0.077	0.068	0.449	0.197	0.211	0.239	0.421	0.483
SMT	0.179	0.065	0.464	0.453	0.121	0.412	0.212	0.473
4-System combination	0.262	0.147	0.605	0.560	0.355	0.639	0.630	0.646
w/learned fixed cutoff	0.260	0.149	0.580	0.534	0.309	0.639	0.644	0.784
Primary run (as submitted)	0.273	**0.156**	**0.637**	**0.605**	**0.420**	**0.687**	**0.706**	**0.791**

4.5.1.1 Query Expansion

Table 38 presents the MQWV scores of the CLIR system with query expansion using word2vec embeddings for the Swahili and Lithuanian EVAL text systems. For Swahili, we observe small gains in MQWV for UMDNMT and UMDSMT systems with query expansion when applied to the entire query set. This might be explained by the poor quality of MT systems for Swahili, which improved query expansion cannot overcome. For Lithuanian, we see larger improvements in MQWV across all three MT systems.

Table 38. Effect of Query Expansion Using word2vec Model on Swahili and Lithuanian EVAL Text Systems (MQWV).



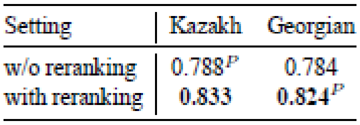
	Swahili			Lithuanian		
Settings	EdiNMT	UMDNMT	UMDSMT	EdiNMT	UMDNMT	UMDSMT
w/o expansion	0.1980	0.1769	0.1934	0.3201	0.3725	0.4170
with expansion	0.1965	0.1779	0.1964	0.3462	0.4052	0.4542

4.5.1.2 XLM-R Reranking

Neural reranking has shown great potential when it comes to improving the retrieval in a typical retrieve-and-rerank pipeline. Here, we test the same hypothesis in the CLIR setting by constructing a pipeline that uses a base CLIR ranking model followed by a neural reranking model initialized with XLM-R embeddings. Our final CLIR submissions included a combination of several of these retrieve and rerank pipelines, each using a different base ranking model. Table 39 shows the effect of reranking on the Kazakh and Georgian text submissions (P denotes primary sub- missions made during respective evaluation). For Kazakh, we observe significant improvements in our contrastive submission with reranking in comparison with our primary run (which did not have the reranking component). For parity, we use the same set of six base CLIR systems for both the primary and contrastive submissions. Based on these results, we use reranking for our Georgian primary submission and saw that reranking also helped CLIR performance in that instance.

We conduct per-query analysis to see the overall effect of reranking on the Kazakh text submission as shown in Figure 21. In general, improvements from reranking are greater in magnitude compared to the cases (queries) in which we observe score degradation. This generates a positive net effect of reranking on CLIR effectiveness.

Table 39. Comparing EVAL Text Systems with and without Reranking (MQWV).



Setting	Kazakh	Georgian
w/o reranking	0.788^P	0.784
with reranking	**0.833**	$\mathbf{0.824}^P$

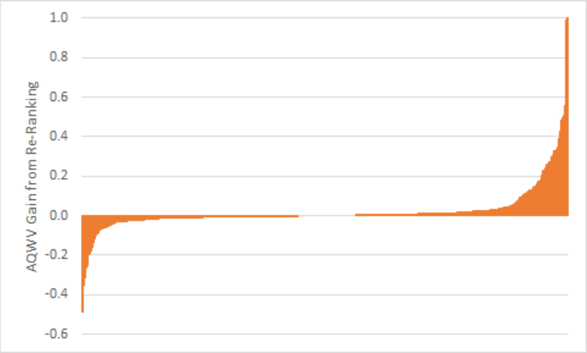


Figure 21. Per-query Comparison of Kazakh Text Primary (without reranking) vs. Contrastive (with Reranking) Systems.

Histogram shows the difference for each query, with queries sorted in increasing order of that difference (positive values indicate a preference for the Contrastive system on that query).

4.5.1.3 Reranking with PACRR and POSIT

We first use the Indri[58] system which combines query likelihood with Dirichlet Smoothing [101] to pre-select documents from the collection. To build the training dataset, we randomly sample one negative example from the documents returned by Indri for each positive example in the returned list. The model is then trained with a binary cross-entropy loss. On the validation and testing sets, we then use our prediction scores to rerank the documents returned by Indri.

4.5.1.4 Extra Features

Following the work in [160], [161], [107], we compute the final relevance score using a linear model to combine the model output with the following set of extra features:

- Indri score using the LM approach to IR
- percentage of query terms with an exact match in the document, including both the regular percentage and IDF weighted percentage
- percentage of query term bigrams matches in the document

[58] www.lemurproject.org/indri.php

4.5.1.5 Cross-lingual Word Embeddings

We apply the supervised iterative Procrustes approach [162], [163] to align two pretrained monolingual fastText [87] word embeddings using the MUSE implementation.[59] To build the bilingual dictionary, we use the translation pages of Wiktionary.[60] For Swahili, Tagalog and Somali, we build training and testing dictionaries as follows: **Swahili** - 5,301 training words/ 1,326 testing words; **Tagalog** - 7,088 training words/1,773 testing words; **Somali** - 7,633 training words/1909 testing words. We then learn the cross-lingual word embeddings from Swahili to English, from Tagalog to English, and from Somali to English, bringing all three languages into the same word embedding space.

4.5.1.6 Baselines

For traditional CLIR approaches, we use query translation and document translation with the Indri system. For query translation, we use Dictionary-Based Query Translation (DBQT) and PSQ. For document translation, we use SMT and NMT. Specifically, for SMT, we use the Moses system [128] with word alignments using mGiza and 5-gram KenLM LM [129]. For NMT, we use S2S model with attention [164], [165] implemented in Marian [43].

For deep relevance ranking baselines, we investigate recent SOTA models including PACRR, PACRR-DRMM, and POSIT-DRMM. These models and our methods all use an SMT-based document translation as input.

Table 40 shows the result on EN->SW and EN->TL where we train and test on the same language pair.

4.5.1.7 Performance of Baselines

For query translation, PSQ performs better than DBQT thanks to its use of a weighted alternative to translate query terms. PSQ also does not limit results to the fixed translation from the dictionary as in DBQT. For document translation, we find that both SMT and NMT perform similarly to PSQ. The effectiveness of different approaches depends on the language pair (e.g., PSQ for EN->SW and SMT for EN->TL), which aligns with findings in [166] and [167]. In our experiments with deep relevance ranking models, we all use SMT and PSQ because they have strong performances in both language pairs.

4.5.1.8 Zero-Shot Transfer Learning

Table 41 shows the result for a zero-shot transfer learning setting where we train on EN->SW + EN->TL and directly test on EN->SO without using any Somali relevance labels.

This transfer learning delivers a 1-3 MAP improvement over PSQ and SMT. This illustrates a promising approach for boosting performance through the use of relevance labels from other language pairs.

4.5.1.9 Aligning Multilingual Contextual Embeddings

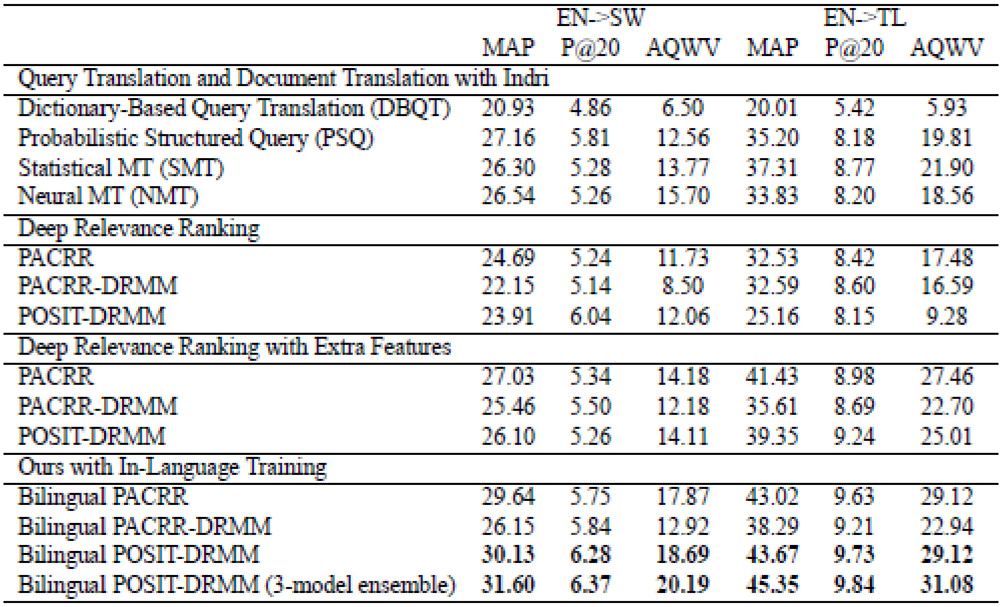
We evaluate our methods on several CLIR collections, sourced from both the IARPA MATERIAL program [168] and the CLEF 2000 - 2003 ad-hoc retrieval collections [169].

[59] http://github.com/facebookresearch/MUSE

[60] https://www.wiktionary.org/

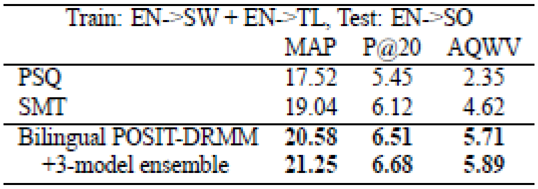
In particular, we use English (EN) queries for all collections, Farsi (FA) and Kazakh (KK) documents from MATERIAL and German (DE) and Finnish (FI) documents from CLEF.

Table 40. Test Set Results on English to Swahili and English to Tagalog.



	EN->SW			EN->TL		
	MAP	P@20	AQWV	MAP	P@20	AQWV
Query Translation and Document Translation with Indri						
Dictionary-Based Query Translation (DBQT)	20.93	4.86	6.50	20.01	5.42	5.93
Probabilistic Structured Query (PSQ)	27.16	5.81	12.56	35.20	8.18	19.81
Statistical MT (SMT)	26.30	5.28	13.77	37.31	8.77	21.90
Neural MT (NMT)	26.54	5.26	15.70	33.83	8.20	18.56
Deep Relevance Ranking						
PACRR	24.69	5.24	11.73	32.53	8.42	17.48
PACRR-DRMM	22.15	5.14	8.50	32.59	8.60	16.59
POSIT-DRMM	23.91	6.04	12.06	25.16	8.15	9.28
Deep Relevance Ranking with Extra Features						
PACRR	27.03	5.34	14.18	41.43	8.98	27.46
PACRR-DRMM	25.46	5.50	12.18	35.61	8.69	22.70
POSIT-DRMM	26.10	5.26	14.11	39.35	9.24	25.01
Ours with In-Language Training						
Bilingual PACRR	29.64	5.75	17.87	43.02	9.63	29.12
Bilingual PACRR-DRMM	26.15	5.84	12.92	38.29	9.21	22.94
Bilingual POSIT-DRMM	**30.13**	**6.28**	**18.69**	**43.67**	**9.73**	**29.12**
Bilingual POSIT-DRMM (3-model ensemble)	**31.60**	**6.37**	**20.19**	**45.35**	**9.84**	**31.08**

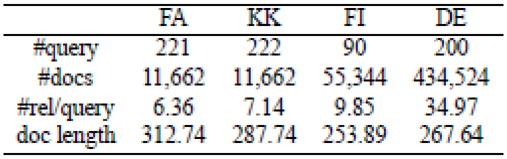
Table 41. Zero-Shot Transfer Learning on English to Somali Test Set.



	Train: EN->SW + EN->TL, Test: EN->SO		
	MAP	P@20	AQWV
PSQ	17.52	5.45	2.35
SMT	19.04	6.12	4.62
Bilingual POSIT-DRMM	**20.58**	**6.51**	**5.71**
+3-model ensemble	**21.25**	**6.68**	**5.89**

We combine all DE or FI test collections from the 2000-2003 CLEF evaluations. For the English CLEF queries, we concatenate the *title* and *description* fields from each of the CLEF topics, as in [170], [171]. For both MATERIAL languages, we use their corresponding DEV sets as our document collections, with queries for that language from MATERIAL QUERY PACK 1. The queries provided with the MATERIAL collections include a conjunctive combination of lexical and conceptual clauses, whereas the queries formed from the CLEF topics include only a single conceptual clause. To limit the effect of this difference on our processing, we flatten the MATERIAL queries by treating the words in each MATERIAL query as a single conceptual clause, thus rendering the MATERIAL and CLEF queries comparable. Results for our CLIR collections are in Table 42.

Table 42. CLIR Collections Statistics.



	FA	KK	FI	DE
#query	221	222	90	200
#docs	11,662	11,662	55,344	434,524
#rel/query	6.36	7.14	9.85	34.97
doc length	312.74	287.74	253.89	267.64

For training and evaluation, we split our query sets evenly into five (disjoint) folds for a five-fold cross-validation setup, where we train our rerankers on four folds, and then test on the fifth. For all of our experiments, we use the P@20, nDCG@20 [172] and MAP measures, following previous work on ad-hoc reranking.

The two alignment methods use different heuristics for selecting word alignments and extracting embeddings: in the technique from [110] only the last sub-word in each word is retained and used to calculate loss, and word alignments used as input are generated using bitext tokenized with an mBERT tokenizer in which all sub-words in each word have been concatenated into a single token. However, awesome-align extracts word alignments in the training process as part of its objective, and two words are considered aligned if any of their sub-words are aligned.

BERT has been shown to learn a classical NLP pipeline [173], [174], with semantic information concentrated closer to the top most layers [175]. As a result, awesome-align uses embeddings from the eighth layer of mBERT for calculating word alignments. By contrast, the technique from [110] aligns the 12th and final layer. This is most widely adopted as the “embeddings” extracted from these LMs.

For both alignment methods, we use the same values for hyperparameters as the original papers use. We also keep the hyperparameters from the original CEDR paper and train each of our reranking models on an NVIDIA RTX 3090 graphic processing unit (GPU).

Table 43 shows the results of our experiments with all of the CEDR models, using either Vanilla mBERT (VBERT) or fine tuning with one of the two alignment methods, along with our PSQ baseline without reranking. We use a two-tailed paired *t*-test ($p < 0.05$) for significance testing. **Bold** indicates the best result on each measure, * indicates significance between mBERT and the marked alignment.

In the **EN-DE** language pair, we improve over the baseline by all measures in both zero-shot alignment and the awesome-align settings. In **EN-FI**, we improve only on P@20 for the zero- shot alignment setting, while for the **EN-KK** language pair, we report worse performance than the baseline across all measures. Meanwhile, we observe in **EN-FA** an improvement over the baseline via awesome-align in only one of our measures, P@20. This discrepancy between measures shows that our neural models achieve better precision by retrieving more of the relevant documents in the collections (P@20), but this may place some other relevant documents lower in rank.

Table 43. Results in German, Finnish, Kazakh, and Farsi.

Language	Method	Embeddings	MAP	P@20	NDCG@20
German (EN-DE)	PSQ		0.3128	0.3043	0.4072
	VBERT	mBERT	0.3138	0.3316	0.4473
		awesome	0.3562*	0.3537*	0.4865*
	DRMM	mBERT	0.3429	0.3441	0.4744
		awesome	0.3616	**0.3739***	0.5040*
	K-NRM	mBERT	0.3460	0.3507	0.4751
		awesome	**0.3674***	0.3704*	**0.5051***
	PACRR	mBERT	0.3218	0.3263	0.4505
		awesome	0.3506*	0.3550*	0.4858*
Finnish (EN-FI)	PSQ		0.3075	0.1880	0.4097
	VBERT	mBERT	0.2918	0.1933	0.3817
		awesome	0.3225	0.2173*	0.4194
	DRMM	mBERT	0.2988	0.2027	0.3936
		awesome	0.3165	**0.2200***	0.4200
	K-NRM	mBERT	0.3129	0.1940	0.4020
		awesome	**0.3374**	0.2167*	**0.4348**
	PACRR	mBERT	0.3013	0.1947	0.3547
		awesome	0.3345*	0.2127	0.4211
Kazakh (EN-KK)	PSQ		**0.5210**	**0.1756**	**0.6103**
	VBERT	mBERT	0.3119	0.1640	0.4147
		awesome	0.3076	0.1621	0.4112
	DRMM	mBERT	0.3126	0.1591	0.4135
		awesome	0.3114	0.1633	0.4154
	K-NRM	mBERT	0.3269	0.1640	0.4333
		awesome	0.3270	0.1625	0.4310
	PACRR	mBERT	0.3018	0.1601	0.4063
		awesome	0.3104	0.1616	0.4106
Farsi (EN-FA)	PSQ		**0.3883**	0.1413	**0.4784**
	VBERT	mBERT	0.2795	0.1637	0.3967
		awesome	0.2897	0.1658	0.4103
	DRMM	mBERT	0.2896	0.1626	0.4281
		awesome	0.2796	0.1638	0.4016
	K-NRM	mBERT	0.2963	0.1587	0.4084
		awesome	0.3057	**0.1683***	0.4261
	PACRR	mBERT	0.2835	0.1649	0.3952
		awesome	0.2990	0.1669	0.4321

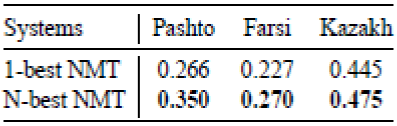
Our results suggest that awesome-align is beneficial for our CLIR task. In both the EN-DE and EN-FI language pairs, alignment leads to improvements over mBERT across several types of rerankers and measures, and for Finnish the use of alignment results offers consistent improvement over the baseline. By contrast, the reranker with Vanilla mBERT fails to rerank documents better than our first-stage PSQ retrieval. In the EN-FA language pair, the awesome-align setting improves over the Vanilla mBERT setting in the P@20 measure, but despite improvement in the other measures, they do not outperform the first-stage retrieval. This means that the aligned setting does not perform worse than our zero-shot aligned setting.

The disparity in performance gains from alignment when compared to the PSQ baseline is to be expected as a result of the low-resource nature of Farsi and Kazakh: mBERT has been shown to perform less well on LRLs for which less text for pretraining is available [176]. As such, the mBERT contextualized embeddings for these languages may be poorer in the zero-shot setting. Some of the other performance differences may also be explained by the selected languages' linguistic distance from English. For example, German is the language most linguistically similar to English, and it achieves the best zero-shot performance. Kazakh and Farsi, meanwhile, both use different scripts than English and do not perform as well in this cross-lingual setting.

4.5.4.10 *N*-best MT

Translation ambiguity is a key challenge for cross-lingual CLIR systems. NMT systems are usually optimized to produce fluent output, but this affects the recall of the CLIR systems. This is particularly true while working with 1-best translation output produced by an NMT system, as it may lack the coverage of required query terms. To avoid this, we make use of the *n*-best translation alternatives to provide better coverage. While we can produce these alternatives at the sentence-level or token-level, we opt for token-level as it promotes more diverse outputs. Table 44 shows the effect on CLIR of using *n*-best translation alternatives as opposed to 1-best with UMDNMT base system. We observe that the *n*-best approach consistently outperforms 1-best in Pashto, Farsi and Kazakh EVAL text systems.

Table 44. Comparing CLIR Systems using 1-best and N-best UMDNMT (MQWV on EVAL Text).



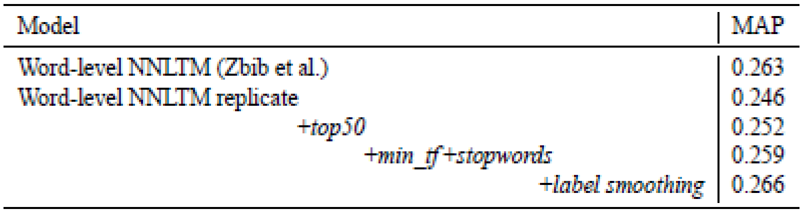
Systems	Pashto	Farsi	Kazakh
1-best NMT	0.266	0.227	0.445
N-best NMT	**0.350**	**0.270**	**0.475**

4.5.1.11 NNLTM

We experiment with changing several hyperparameters associated with the NNLTM that might affect the retrieval performance. The number of contextualized translations stored is increased from 10 to 50 (*top 50*). In addition to that, we remove samples from training that have either English *stopwords* as the target or occur less than five times (*min_tf*).

We also employ a regularization technique called *label smoothing* [39] which prevents the model from becoming overconfident in its predictions. This technique involves smoothing the one hot target labels with a uniform distribution over the target vocabulary size. These smoothed target labels are then used to train the model. We use 0.1 as the value for label smoothing parameter based on [39]. Results from all of our changes in the original NNLTM model are summarized in Table 45.

Table 45. Results of NNLTM Models on Swahili EVAL used in Zbib et al.[2].



Model	MAP
Word-level NNLTM (Zbib et al.)	0.263
Word-level NNLTM replicate	0.246
+top50	0.252
+min_tf+stopwords	0.259
+label smoothing	0.266

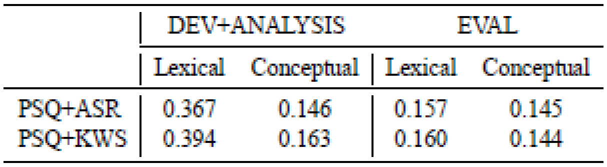
For ease of comparison, we ran these experiments on the same setup as the one used in [2] - specifically, the IARPA EVAL set, with query sets Q1 and Q3 for Swahili.[61] The model in [2] is compared with our replicated model which uses the same configuration. The models do differ in the training data, however, which we believe causes the differences between the two. Specifically, we do not have access to the LORELEI [177] Swahili data used to train the original model. As our replicated model outperforms the original, however, suggesting that our model would outperform the original on identical training data as well.

4.5.1.12 Keyword Spotting

This section details the effect of different retrieval methods for Swahili, using MAP as the evaluation measure. MAP is equivalent to Mean Reciprocal Rank any time only one relevant document exists, as is often the case for the DEV collection in which 64 of the 126 queries with any relevant documents have precisely one.

The effect of the two retrieval methods, PSQ+ASR and PSQ+KWS, is shown by query type (for non-conjunctive queries) in Table 46. MAP for lexical queries increases significantly when using KWS over that of 1-best ASR. For conceptual queries, gains are apparent on the DEV set, but MAP is essentially unchanged on the larger EVAL set. We therefore conclude that PSQ+KWS is the preferred approach for both basic query types.

Table 46. MAP Scores of Different Query Types for Swahili Test Collection.



	DEV+ANALYSIS		EVAL	
	Lexical	Conceptual	Lexical	Conceptual
PSQ+ASR	0.367	0.146	0.157	0.145
PSQ+KWS	0.394	0.163	0.160	0.144

Table 47 summarizes the overall improvements on the two Swahili test collections. We observed that switching from ASR to KWS yielded a statistically significant improvement and that also adding conjunction processing using the geometric mean (GeoMean) yielded yet another statistically significant improvement. Moreover, the net improvement from the combination of these two changes is substantial: 12% (relative) on the larger EVAL collection, and 21% on the small DEV collection. In the table, x, y and z denote statistically significant improvements over ASR, KWS and ASR+GeoMean, respectively, using a two-tailed Wilcoxon signed rank test with $p < 0.05$.

[61] We only use Swahili results as it is the only Eval set used in [2].

Table 47. MAP for all Queries.

Systems	Swahili DEV	Swahili EVAL	Tagalog DEV
PSQ+ASR	0.288	0.165	0.388
PSQ+KWS	0.303x	0.168	0.406x
PSQ+ASR+GeoMean	0.329x	0.181x	0.417x
PSQ+KWS+GeoMean	0.349xyz	0.184xyz	0.458xyz
PSQ+Manual Transcription			0.485
Manual Translation & Transcription			0.512
Manual Translation & Transcription+GeoMean			0.513

We didn't have separate DEV and EVAL sets for Tagalog, but as Table 47 shows, we could observe similar trends on the one relatively small Tagalog collection as we saw with Swahili. Statistically significant improvements result from each change and the net improvement from the two together is 18% (relative). We therefore concluded the choices made for Swahili are reasonable choices for Tagalog as well. About two-thirds (250) of the 387 Tagalog queries that have any relevant documents at all have only one relevant document. Our best Tagalog result (a MAP of 0.458) roughly corresponds to placing a single relevant document at rank 2 (the Mean Reciprocal Rank for a system that always placed the first relevant document at rank 2 would be 0.5). This is credible performance for an LRL as is the comparable value for our similarly sized Swahili DEV set, which equates to roughly rank 3. We believe these values are likely good enough to be useful in practical applications.

For the Tagalog test collection, we also have manual 1-best transcription and manual 1-best translation available. As Table 47 shows, using these 1-best manual processes yields a MAP of 0.513 for Tagalog, which is only 12% (relative) above our best present Tagalog result. While we note that 1-best transcription and translation are not an upper bound on what can be achieved by systems with good modeling of translation ambiguity, we take this small gap as further confirmation that our Tagalog system is yielding credible results. As Figure 22 shows, one possible source of the difference is that the expected term count underestimates the correct term count more often than it overestimates this count (as measured on the 1-best translation). Averaging over all terms in the collection, the mean absolute error of the expected counts is 1.727.

4.5.1.13 Early and Late Fusion

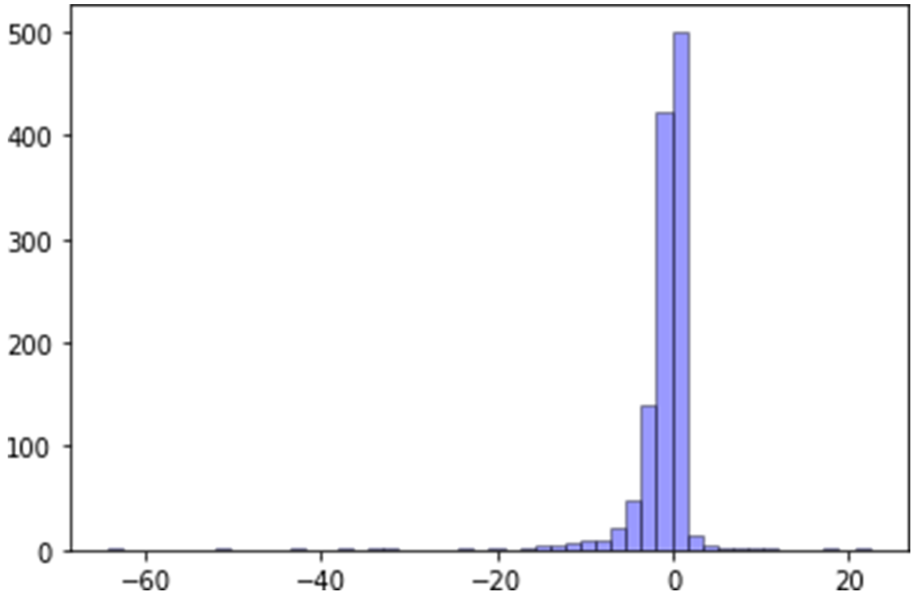
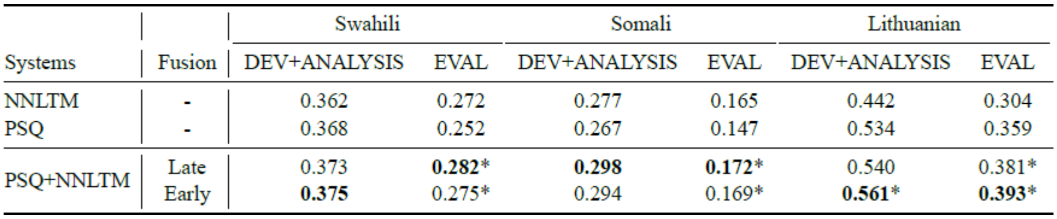


Figure 22. Difference between Expected and Actual Term Count.

To test the performance of early and late fusion techniques, we combined the enhanced NNLTM model with the PSQ non-contextual translations using both the fusion approaches. The enhanced model that achieved a MAP of 0.266 in the Table 45 on the Q1+Q3 query sets achieved a MAP of 0.272 on our EVAL set with Q2+Q3 (that is, using the same documents, but some different queries). The results for the individual systems and their combinations are shown in Table 48.
Bold indicates best results per column, * indicates statistically significant improvement over both single systems in the combination. Two-tailed Wilcoxon signed rank test with p<0.01 is applied.

Table 48. MAP of the NNLTM and PSQ Models on the MATERIAL DEV and EVAL Collections.



		Swahili		Somali		Lithuanian	
Systems	Fusion	DEV+ANALYSIS	EVAL	DEV+ANALYSIS	EVAL	DEV+ANALYSIS	EVAL
NNLTM	-	0.362	0.272	0.277	0.165	0.442	0.304
PSQ	-	0.368	0.252	0.267	0.147	0.534	0.359
PSQ+NNLTM	Late	0.373	**0.282****	**0.298**	**0.172***	0.540	0.381*
	Early	**0.375**	0.275*	0.294	0.169*	**0.561***	**0.393***

We observed the PSQ system outperformed the NNLTM model on Lithuanian, which is the language with the most available resources among the tested languages. NNLTM outperformed PSQ on Swahili and Somali, except on the very small Swahili DEV+ANALYSIS set. The late fusion combination on each EVAL collection significantly outperformed the individual under-lying systems, with the largest differences achieved on the Lithuanian. In the case of the Lithuanian and Swahili DEV+ANALYSIS sets, early fusion combination further improves these results.

The strong performance of both combination methods confirms our assumption that the noise resulting from working with the *n*-best possible translations can be effectively overcome by combining multiple such systems.

4.5.1.13.1 Late Fusion

Combining several individual systems to improve performance has been successfully used in different research areas and applications, including ML [178], speech recognition [179], and IR [112]. For late fusion (i.e., combination of ranked lists) to be helpful, the combined systems need to be both well performing (although not necessarily best performing) and diverse. Late fusion proved beneficial in almost all of our experiments, but in building a large number of configurations (for some languages, more than 1,000 different setups), we encountered new research problems. For example, *which systems should be combined to achieve the best performance*? To assess this, we began with baselines of 1) the single best performing system, and 2) the combi- nation of all available systems. In the end, the best performance was achieved by combining a relatively small number of sufficiently diverse systems, each of which performed reasonably well in its own right. Manual election of the systems, using these criteria and a human conceptual knowledge of the systems, was used to generate the primary run in OP1 and OP2. Though we also experimented with automatic clustering methods to identify systems returning similar results this strategy can sometimes place many of the best performing systems in a single cluster, and those cases lead to suboptimal results.

Comparison of systems submitted during the Farsi evaluation on EVAL collections are shown in Table 49 for text and Table 50 for speech. Manual selection of six systems clearly outperforms the system that was the single best performing system on the development collection, and also outperforms a selection of four systems on text. On speech, similar results are seen, with manual selection of six systems also performing better than the single system that had performed best on the development collection, but also manually selected eight systems on speech. However, the overall best score on speech was achieved using 12 systems selected using automatic clustering. In both tables, the blue line is the primary run submitted, and the best achieved results are in bold. The cutoff states the target cutoff value and also the cutoff approach used (’A’ stands for average of STO, QST and Fixed-rank cutoff, ’F’ stands for fixed-rank cutoff and ’Q’ stands for the average of QST and fixed-rank cutoff.

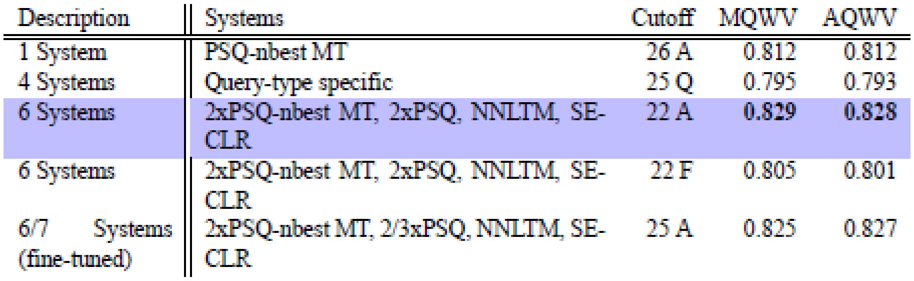
In addition to the number of systems, we have also experimented with query specific strategies for text (see line 2 in Table 49). In this case, four systems were run on each query and combined, choosing the systems systematically by query type, with the goal of manually optimizing the selected systems for each query type. However, this strategy did not yield improvements’

4.5.1.13.2 Cutoffs

We experimented with several strategies for selecting how many documents to return from the top of the ranked list for each query. For most of our primary runs in OP1 and OP2, we used an average of STO, QST and fixed-rank prediction; we also limited the maximum number of documents to be returned as a means of false alarm control. This method led to nearly optimal results on text, especially for the OP2 languages for which larger DEV collections were available. The sizes of the DEV and EVAL sets were nearly the same for text, which obviated the need for scaling of parameters. This was not the case for speech, however, where the size of the DEV collection was still considerably smaller than the size of the EVAL collection. The STO and fixed-rank cutoff parameters for speech thus needed to be adjusted, incurring a degree of estimation error. In OP1 and OP2, it was possible to use experience from previous languages to inform our parameter choices, allowing us to mitigate estimation errors somewhat. These observations are visible in Table 49 and Table 50 for Farsi text and speech, respectively. The AQWV score for text is almost equal to the corresponding MQWV scores, indicating the cutoffs are close to optimal. However, the differences between AQWV and MQWV scores are considerably larger on speech. Examining the corresponding Detection Error Tradeoff (DET) curves (not shown), we see the number of returned documents was consistently too large in the case of speech. Comparing lines 3 and 4 in Tables 49 and 50 also provides a comparison between using a fixed-ranked cutoff and averaging cutoffs on a per-query basis over the three threshold selection methods.

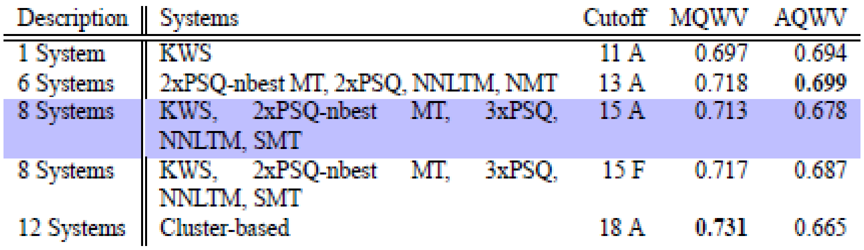
Table 49. Farsi Text Results Submitted during the Evaluation.

The purple line is the primary run submitted.



Description	Systems	Cutoff	MQWV	AQWV
1 System	PSQ-nbest MT	26 A	0.812	0.812
4 Systems	Query-type specific	25 Q	0.795	0.793
6 Systems	2xPSQ-nbest MT, 2xPSQ, NNLTM, SE-CLR	22 A	**0.829**	**0.828**
6 Systems	2xPSQ-nbest MT, 2xPSQ, NNLTM, SE-CLR	22 F	0.805	0.801
6/7 Systems (fine-tuned)	2xPSQ-nbest MT, 2/3xPSQ, NNLTM, SE-CLR	25 A	0.825	0.827

Table 50. Farsi Speech Results Submitted during the Evaluation.



Description	Systems	Cutoff	MQWV	AQWV
1 System	KWS	11 A	0.697	0.694
6 Systems	2xPSQ-nbest MT, 2xPSQ, NNLTM, NMT	13 A	0.718	**0.699**
8 Systems	KWS, 2xPSQ-nbest MT, 3xPSQ, NNLTM, SMT	15 A	0.713	0.678
8 Systems	KWS, 2xPSQ-nbest MT, 3xPSQ, NNLTM, SMT	15 F	0.717	0.687
12 Systems	Cluster-based	18 A	**0.731**	0.665

Line 4 uses a fixed-ranked cutoff, while the rest of the experiments use averages over two or three methods. In text, the average strategy simply leads to better results for both MQWV (0.829 vs. 0.805) and AQWV (0.828 vs. 0.801).

Surprisingly, the results for the fixed-ranked cutoff for speech are almost the same as the averaging method for MQWV (0.713 vs. 0.717) and are actually better for AQWV (0.678 vs. 0.687). This further illustrates the challenge of optimal parameter estimation for speech.

The pattern for Kazakh is similar; using the average of three cutoff strategies for Kazakh and the MQWV for the primary run dropped from 0.807 to 0.800 AQWV in text and from 0.742 MQWV to 0.702 AQWV in speech. Moreover, in Kazakh, there was only as small difference between the MQWV values of the average of three systems and the fixed cutoff value—it dropped from 0.807 MQWV for the average for text to 0.799 MQWV to fixed-ranked cutoff for text and from 0.742 to 0.740 for speech. Due to the small performance difference between average cutoff and fixed-ranked cutoff, we used the average of only STO and KST cutoffs as the cutoff strategy for the primary run in Georgian. However, it achieved similar results: 0.824 MQWV, 0.821 AQWV results for the primary text system and 0.791 MQWV and 0.784 MQWV for speech.[62]

[62] We accidentally submitted the wrong run as the primary speech run during the evaluation and calculated these values for the intended submission after the evaluation.

The same run achieved 0.820 MQWV score for text and 0.797 for speech. Though we don't get much improvement using the averages of the three strategies (STO, QST and fixed-ranked cutoff) against the fixed-ranked cutoff, it outperforms it quite steadily and is quite robust. As a result, we believe that it is a good choice for the primary run.

4.5.1.13.3 Sprints

Figure 23 illustrates a typical case for a sprint, in which the goal was to build the best possible system in a highly constrained time period. In this case, the language was Pashto and the time period was a single week. Near-peak results were obtained for text on the second day, using PSQ, with near-peak MT systems (as measured by their utility for CLIR) becoming available on the fourth day. Near-peak results for speech were also obtained on the fourth day, again using PSQ. The apparent decline in the best text results on the fifth day results from measurement limitations of the small DEV set; systems were being tuned on DEV+ANALYSIS, and the day five results are were obtained using system combination. It should be noted that the EVAL collection was not available during these sprint exercises, so the small DEV collections were used both in training and for evaluation. While these results are useful as a measure of relative improvements over time, the absolute MQWV values are not necessarily indicative of the results on EVAL. For comparison, our primary run on Pashto EVAL, submitted months later, achieved an AQWV of 0.509. Moreover, our principal focus during these sprint evaluations was on ranking (not cutoffs); this is why MQWV is reported rather than AQWV. Even with these caveats, our sprint results indicate that rapid development of reasonably effective systems is possible.

Although sprints had not been an original design element of the program, we found them useful for getting started on a new language, as by the end of a sprint we had fairly complete systems that could then serve as a basis for subsequent component development. On the other hand, the program's announcements of new languages had not been timed with sprints in mind, so in the case of Pashto, we found ourselves conducting a sprint during the end-of-year holiday period, when only a minority of the research team was actually working from their usual locations.

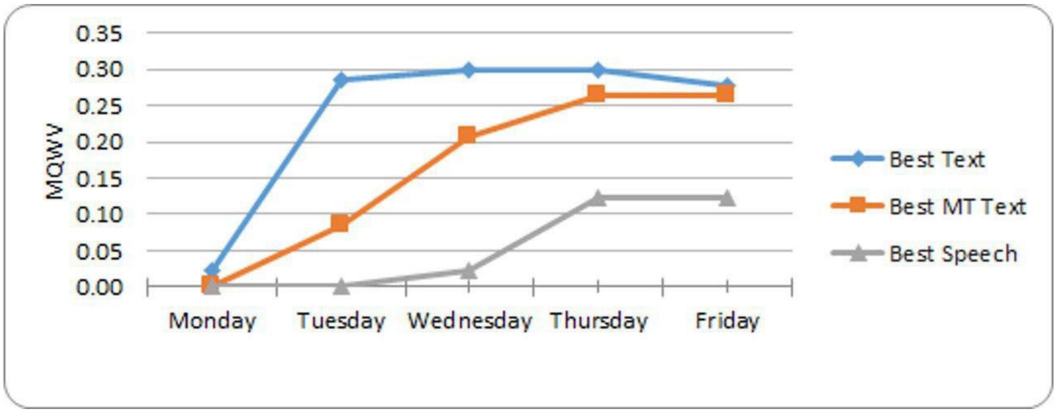


Figure 23. Best MQWV per Day on DEV - Pashto Five-day Sprint.

4.5.2 Technical Insights

Here we draw out larger lessons from our work on CLIR in MATERIAL.

4.5.2.1 The Importance of Regular Orthography

Perhaps the most surprising challenge in the MATERIAL program was our inability to model the way Somali is written. Looking back, it's clear this was a blind spot resulting directly from our experience with predominantly high resource, well standardized languages. Broadly speaking, high resource languages are naturally well standardized because they must be - the cultural, commercial and political factors that resulted in those resources being produced are the same factors that resulted in the written forms of those languages being well standardized. Somali is neither high resource nor well standardized, and it may be emblematic of a class of challenges we are likely to encounter in other languages. Of course, variation is not unique to written language, and indeed the problem of regional and dialectal variation is better studied in speech than in text at present, at least among those building language technologies. In speech, a classic approach to deal with variation is to build on mixture models and that precedent may offer a useful starting point for text as well. Highlighting this challenge is a contribution of the MATERIAL program and knowing of the challenge will help us all to properly prioritize the effort that will be needed to address it.

4.5.2.2 The Value of a Configurable Pipeline

To support experimentation, we opted to use a fully configurable pipeline, meaning that each of the setups described in the technical description can be modified via the configuration file. Some of the configurations were set directly using an option (e.g., the cutoff or combination type) or through the path to the file in the directory structure. Unique identification of the directory structure then enables using different text and speech pre-processing setups, speech processing and MT systems, or indexing parameters (e.g., using stemming). Our fully configurable pipeline allowed us to script the execution of a large number of experiments, run small modifications of the experiments (e.g., different translation types), and to trivially apply a given configuration to another dataset. We gained even more from this approach by using unified names that uniquely defined the experiments but were still short enough to be readable. This helped us more easily visualize the experiments and setup, allowing for crucial improvements in the process for selecting the systems for combination. Our pipeline was designed to support experimentation - and especially reproducibility. We also designed the system to actively avoid possible errors and support error trace-back. A concrete outcome of this choice was the intermediate results (e.g., the output of the query analyzer or the output of the rankers) were saved. This proved to be a strong choice for enhancing flexibility. Though efficiency was not a main objective of the MATERIAL program, this design had costs in both processing time and storage requirements. These challenges were exacerbated as the pipeline became more complex, and this led to periodic refactoring. This also led us to prepare a completely separate pipeline for the demo.

4.5.2.3 The Centrality of IR Training Data

At the outset of the MATERIAL program, it was already well recognized that training data was essential for ASR and for MT. Today, we think of IR in the same way; neural transformer models - pre-trained on large text collections and fine- tuned on large IR collections such as Microsoft Machine Reading Comprehension (MS MARCO)[63] [180] - now routinely achieve results far better than those of the best IR systems from just a few years ago. MATERIAL

[63] https://microsoft.github.io/msmarco/

adapted to this emerging opportunity, reconceptualizing the role of IR training data and reallocating resources in response. But the centrality of relevance judgments to training IR systems restricted what could be accomplished because by the time the need had become clear, the relevance judgements had already been bought and paid for. This limitation is also an opportunity, however, as it illustrates what will be needed to take the next step. As we cannot reasonably make an MS MARCO collection for each of the world's languages, we will need to rely on transfer learning that we can fine-tune to the task, context, and peculiarities of specific languages. The question now is how we focus the data on which we will fine-tune. In MATERIAL, the language-specific training data we had was distributionally similar to the EVAL collection on which the summative evaluation was performed. Given a budget and a timeframe, however, is that the data we would want? Not necessarily. That is the next question to be answered.

4.5.2.4 The Limits of Small Training Collections

The early 300-document DEV collections were simply not large enough to reliably predict performance of IR techniques on the EVAL collection, due to the limited number (1̃50) of positive relevance judgments. Larger test collections were needed for formative evaluation, but larger test collections are expensive, and curating them could take longer than allowable in some actual deployment scenarios. In addressing this challenge, we might take a page from our MT and ASR colleagues who also require larger training sets. For IR, we might assemble some large and diverse set of training data by instrumenting a live application (e.g., a Web search engine) for which click streams provide useful implicit feedback, and then use our DEV collection as a probe into that space to find suitable content. Though it might be effective, this might generate rights management issues given the commercial value of such a collection, as well as its potential privacy risks. Some of these could potentially be mitigated by sharing only the relevant data features rather than the entire dataset; some distortion to protect privacy and commercial interests might also be possible. Employing differential privacy might also be a way forward, although high dimensionality and the importance of rare phenomena for IR might pose significant challenges for this approach. Regardless, developing effective methods for generating larger IR training sets is clearly an important research problem going forward, as mining very small DEV collections proved a significant limitation in this research.

4.5.2.5 Leveraging the IR Research Community

The many types of research that were a part of MATERIAL produced quite a complex task and we are now well positioned to consider what the IR community should focus on next. To do that, we also needed to identify the best venue through which to organize and how we could shape the task to attract the expertise and investment required while also maximizing the chances of success.

One possible focus solution is Text Retrieval Conference (TREC), Neural CLIR, and a new test collection—and indeed such a NeuCLIR track is being proposed to TREC 2020 with just that structure. Alternatively, MediaEVal[64] would be an excellent venue to push on speech retrieval research. The Forum for Information Retrieval Evaluation[65] (FIRE) in India would also be an excellent place to push on LRLs.

[64] https://multimediaeval.github.io/

[65] http://fire.irsi.res.in

4.5.2.6 Modeling Ambiguity

The representation of uncertainty is central to all of IR because our interpretation of human language - both in the query and in the documents - is necessarily imperfect. This limitation is ultimately unavoidable, because ambiguity is inherent in human language. CLIR and to an even greater extent Cross-Language Speech Retrieval (CL-SR), stresses our ability to model meaning in computationally useful ways, but we have advanced our understandings through MATERIAL. First, we now have an elegant unification of perspectives on accommodating translation and recognition ambiguity; we used alternate recognition hypotheses in the same way that we use alternative translation hypotheses. Said another way, what we first called PSQ back in 2003 is the Swiss army knife of cross-modal and CLIR. This perspective is elegant in both its relative computational simplicity and the opportunity it provides for early fusion. The cost of this elegance, however, is that we lose the ability to model the sequential aspects of human language. We clearly see from our work with phrase search, the sequential dependence model, and the neural transformer models that the sequence of terms encodes meaning that cannot be obtained from isolated terms. We are, however, still early in our thinking about how best to model this sequential evidence. In 2002, Federico proposed a very limited version of this when he suggested that term pairs be modeled, but (because of translation divergences) as unordered rather than ordered sequences [181]. The self-attention in transformers gives us an alternative starting point for modeling sequences, but that is one in which both association and order are encoded. With enough training data, transformers could learn what is important for CLIR, but sufficient training data for this approach is currently out of reach. In the meantime, we need new models that better reflect the structure of the problem; something between Federico's approach and today's transformers, designed from our understanding of what matters in the CLIR realm.

4.6 Summarization

Below, we provide E2E results for every language in the MATERIAL program, as well as experimental results for the components as they were developed.[66] Component subsections are ordered by time period, with each subsection including experimental results corresponding to the development of that component. E2E results for each language can be found within their own subsections.

4.6.1 Unsupervised Approach

Results from our unsupervised approach are reflected in our E2E evaluation scores for the BP languages. We do not include experimental results obtained during development of the unsupervised system.

4.6.2 E2E Evaluation Scores and Reclassification Rates for BP - Languages 1A, 1B and 1S

The summarization system for Swahili (1A), Tagalog (1B), and Somali (1S) relied only on the unsupervised approach. The embeddings used were Glove6B and Glove42B (300 dimensional) [182] for English embeddings and fastText (300 dimensional) [183] for foreign language embeddings. We show the E2E evaluation results and reclassification rates in Tables 51, 52, and 53. In these tables, we provide the breakdown of documents whose labels were changed (or not changed) from the CLIR prediction to the human-assisted prediction using the summary; for example, ”fa-to-tn” refers to documents that were incorrectly predicted as relevant by CLIR (a

[66] These sometimes include results on other corpora, to show the performance of our models as they were trained

"false alarm") but were correctly identified as irrelevant by human workers upon seeing the summary (a "true negative"). A "td" refers to a document that is correctly predicted as relevant, and a "miss" is incorrectly considered irrelevant by human workers. Finally, we also present the AQWV, as in CLIR, after our summaries have been provided to a user (i.e., AQWV for the E2E system).

Table 51. E2E Evaluation Results and Reclassification Rates for Swahili EVAL.

query type	fa-to-tn %	fa-to-fa %	td-to-miss %	td-to-td %	AQWV
All	91.91%	8.09%	67.68%	32.32%	0.085
Lex	90.85%	9.15%	63.71%	36.29%	0.073
Lex:Lex	92.69%	7.31%	71.73%	28.27%	0.067
Concept	90.35%	9.65%	58.62%	41.38%	0.170

4.6.3 Speech Segmentation to Improve Fluency

In this section, we discuss the training and test data we used while developing the speech segmentation models. We also present the experimental results that led us to include the model in our OP1 systems for Lithuanian (2B) and Bulgarian (2S). More detailed experimental results can be found in [184].

Table 52. E2E Evaluation Results and Reclassification Rates for Tagalog EVAL.

query type	fa-to-tn %	fa-to-fa %	td-to-miss %	td-to-td %	AQWV
All	94.35%	5.65%	58.25%	41.75%	0.216
Lex	92.35%	7.65%	53.19%	46.81%	0.240
Lex:Lex	97.06%	2.94%	65.58%	34.42%	0.201
Concept	94.48%	5.52%	54.94%	45.06%	0.209

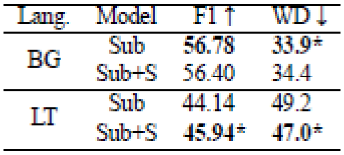
Table 53. E2E Evaluation Results and Reclassification Rates for Somali EVAL.

query type	fa-to-tn %	fa-to-fa %	td-to-miss %	td-to-td %	AQWV
All	85.57%	14.43%	48.12%	51.88%	0.100
Lex	81.30%	18.70%	37.79%	62.21%	0.081
Lex:Lex	84.40%	15.60%	43.47%	56.53%	0.095
Concept	83.75%	16.25%	60.87%	39.13%	0.109
All Speech	85.70%	14.30%	52.45%	47.55%	0.073
All Text	85.33%	14.67%	47.13%	52.87%	0.106

Training data was obtained from Open Subtitles [185] for both Bulgarian (BG) and Lithuanian (LT). To perform extrinsic evaluation of a speech-to-text translation (STTT) pipeline, the ANALYSIS speech collection was used to evaluate performance on downstream tasks (e.g., segmentation quality, MT, and IR). DEV was used to evaluate downstream performance only on IR. We experiment using the ASR component from UMD and EDIN and three MT components - UMD-NMT and UMD-SMT from UMD, and EDI-NMT from EDIN. The IR model is a bag-of-words query model.

For intrinsic evaluation, the models are evaluated on the F-measure of the boundary prediction labels, as well as WindowDiff (WD) [186], a metric that penalizes difference in the number of boundaries between the reference and predicted segmentation over a fixed window. Our models are depicted as Sub and Sub+S where Sub+S utilizes syntactic features like POS tags and dependency labels. Results are shown in Table 54, where +S indicates syntactic features and * indicates statistical significance.

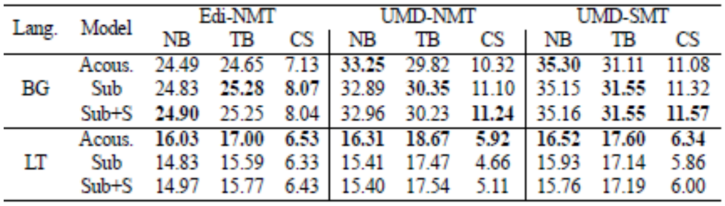
Table 54. Results for Intrinsic Evaluation of Speech Segmentation Models, F1 and WD on ANALYSIS.



Lang.	Model	F1 ↑	WD ↓
BG	Sub	**56.78**	**33.9***
	Sub+S	56.40	34.4
LT	Sub	44.14	49.2
	Sub+S	**45.94***	**47.0***

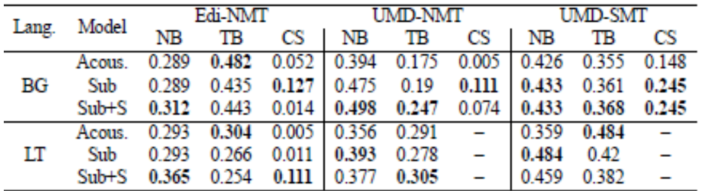
For extrinsic evaluation, Table 55 shows the results of document level BLEU score of the MT output on ANALYSIS. Note that in Tables 55, 56, and 57, "Acous." denotes the baseline performance of the AM of the ASR system. Tables 56 and 57 show results from evaluating the performance of CLIR using AQWV, where +S indicates syntactic features and * indicates statistical significance. In all three tables, NB denotes "news broadcast," TB denotes "topical broadcast," and CS denotes "conversational speech."

Table 55. Experimental Results for Speech Segmentation Usage for Document Level BLEU Scores on ANALYSIS Set.



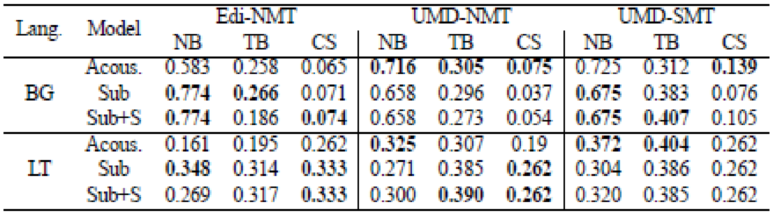
Lang.	Model	Edi-NMT			UMD-NMT			UMD-SMT		
		NB	TB	CS	NB	TB	CS	NB	TB	CS
BG	Acous.	24.49	24.65	7.13	**33.25**	29.82	10.32	**35.30**	31.11	11.08
	Sub	24.83	**25.28**	**8.07**	32.89	**30.35**	11.10	35.15	**31.55**	11.32
	Sub+S	**24.90**	25.25	8.04	32.96	30.23	**11.24**	35.16	**31.55**	**11.57**
LT	Acous.	**16.03**	**17.00**	**6.53**	**16.31**	**18.67**	**5.92**	**16.52**	**17.60**	**6.34**
	Sub	14.83	15.59	6.33	15.41	17.47	4.66	15.93	17.14	5.86
	Sub+S	14.97	15.77	6.43	15.40	17.54	5.11	15.76	17.19	6.00

Table 56. Results on ANALYSIS for Speech Segmentation, AQWV for NB, TB, and CS.



Lang.	Model	Edi-NMT			UMD-NMT			UMD-SMT		
		NB	TB	CS	NB	TB	CS	NB	TB	CS
BG	Acous.	0.289	**0.482**	0.052	0.394	0.175	0.005	0.426	0.355	0.148
	Sub	0.289	0.435	**0.127**	0.475	0.19	**0.111**	**0.433**	0.361	**0.245**
	Sub+S	**0.312**	0.443	0.014	**0.498**	**0.247**	0.074	**0.433**	**0.368**	**0.245**
LT	Acous.	0.293	**0.304**	0.005	0.356	0.291	–	0.359	**0.484**	–
	Sub	0.293	0.266	0.011	**0.393**	0.278	–	**0.484**	0.42	–
	Sub+S	**0.365**	0.254	**0.111**	0.377	**0.305**	–	0.459	0.382	–

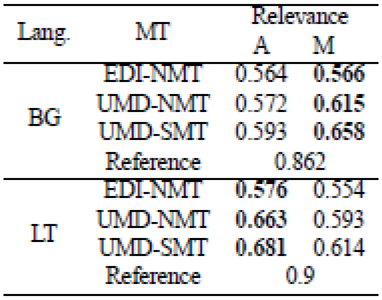
Table 57. Results on DEV for Speech Segmentation, AQWV for NB, TB, and CS.



Lang.	Model	Edi-NMT			UMD-NMT			UMD-SMT		
		NB	TB	CS	NB	TB	CS	NB	TB	CS
BG	Acous.	0.583	0.258	0.065	**0.716**	**0.305**	**0.075**	0.725	0.312	**0.139**
	Sub	**0.774**	**0.266**	0.071	0.658	0.296	0.037	**0.675**	0.383	0.076
	Sub+S	**0.774**	0.186	**0.074**	0.658	0.273	0.054	**0.675**	**0.407**	0.105
LT	Acous.	0.161	0.195	0.262	**0.325**	0.307	0.19	**0.372**	**0.404**	0.262
	Sub	**0.348**	0.314	**0.333**	0.271	0.385	**0.262**	0.304	0.386	0.262
	Sub+S	0.269	0.317	**0.333**	0.300	**0.390**	**0.262**	0.320	0.385	0.262

We also conducted human evaluation to compare this segmentation with acoustically-based segmentation in order to determine the effect on annotators who attempt to determine MT fluency and query relevance. We use YAKE![187] to extract keywords from ANALYSIS documents and present annotators with a keyword and a three-segment passage from a document containing that keyword. These results are in Table 58.

Table 58. Results for Speech Segmentation Usage, Passage-Level Evaluation Comparing Relevance (Sub+S Model (M) and the Acoustic Baseline (A).



Lang.	MT	Relevance	
		A	M
BG	EDI-NMT	0.564	**0.566**
	UMD-NMT	0.572	**0.615**
	UMD-SMT	0.593	**0.658**
	Reference	0.862	
LT	EDI-NMT	**0.576**	0.554
	UMD-NMT	**0.663**	0.593
	UMD-SMT	**0.681**	0.614
	Reference	0.9	

Generally, higher volumes of subtitle data generate consistent improvements on MT and CLIR tasks. Even for LT, which does not have a lot of data, there is still improvement in the document-level CLIR task–this is most visible in the CS genre, which most resembles our subtitle training data. We see improvement (for those conditions where we have appropriate training data) under all neural systems on CS.

4.6.4 E2E Evaluation Scores and Reclassification Rates for 2B and 2S

For Lithuanian (2B) and Bulgarian (2S), the final system used the unsupervised approach and speech segmentation. The embeddings used were Glove6B and Glove42B (300 dimensional) for English embeddings and fastText (300 dimensional) for foreign language embeddings. Tables 59 and 60 contain E2E evaluation results for Lithuanian and Bulgarian respectively; we note that at this time, the E2E evaluation process was experimenting with allowing human annotators the freedom to rate the generated summaries on a scale of relevance, rather than simple binary judgments. Here, we were provided (and we provide) AQWV E2E scores based on what rating (from 1-5) is used as the threshold for "relevant," where 1 is definitely irrelevant and 5 is definitely relevant (in the remaining evaluations through this paper, it is 3). We see a spread of confidence in the relevance of our summaries, with neutrally more documents considered "relevant" if the cutoff is at 2 on a 1-5 scale than if it is at 5.

4.6.5 E2E Evaluation Scores and Reclassification Rates for 2C

For Pashto, the final system used the unsupervised approach only. The embeddings used were Glove6B, Glove42B (300 dimensional), and RoBERTa [188] for English embeddings and fastText (300 dimensional) for foreign language embeddings. We did not have enough training data from Open Subtitles for Pashto to be able to develop and use the speech segmentation component. Table 61 contains E2E evaluation results. Table 62 contains reclassification rates. Note that in these tables we report the proportion of times our final E2E system produced a summary that convinced human workers a ground truth (GT) relevant (GT-Rel) document was irrelevant (Pmiss rel); the same for a GT irrelevant (GT-Irrel) document being marked relevant (Pfa), and the reclassification rates (i.e., our summaries yielded an irrelevant judgment when CLIR predicted the document was relevant) of GT relevant (TD reclass) and irrelevant (FA reclass) documents.

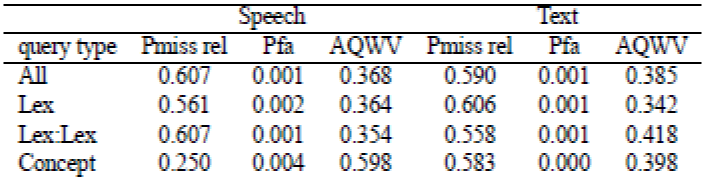
Table 59. E2E AQWV Results for Lithuanian EVAL.

	Speech	Text
Level 2 AMT-E2E	0.587	0.621
Level 3 AMT-E2E	0.560	0.593
Level 4 AMT-E2E	0.547	0.579
Level 5 AMT-E2E	0.438	0.450

Table 60. E2E AQWV Results for Bulgarian EVAL.

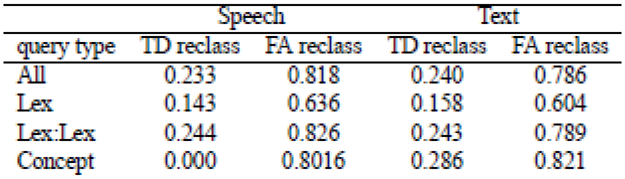
	Speech	Text
Level 2 AMT-E2E	0.622	0.678
Level 3 AMT-E2E	0.596	0.656
Level 4 AMT-E2E	0.587	0.632
Level 5 AMT-E2E	0.432	0.491

Table 61. E2E Results for Pashto EVAL.



	Speech			Text		
query type	Pmiss rel	Pfa	AQWV	Pmiss rel	Pfa	AQWV
All	0.607	0.001	0.368	0.590	0.001	0.385
Lex	0.561	0.002	0.364	0.606	0.001	0.342
Lex:Lex	0.607	0.001	0.354	0.558	0.001	0.418
Concept	0.250	0.004	0.598	0.583	0.000	0.398

Table 62. Reclassification Rates for Pashto EVAL.



	Speech		Text	
query type	TD reclass	FA reclass	TD reclass	FA reclass
All	0.233	0.818	0.240	0.786
Lex	0.143	0.636	0.158	0.604
Lex:Lex	0.244	0.826	0.243	0.789
Concept	0.000	0.8016	0.286	0.821

4.6.6 Supervised Approach - Query Relevance Sentence Selection

In this section, we discuss the training and test data we used while developing the query relevance sentence selection model and present experimental results which led us to include this component in our submitted system for all OP2 languages (Farsi/3S, Kazakh/3C, and Georgian/3B).

The parallel sentence data for training and baselines comes from the BUILD collections of the MATERIAL and LORELEI [177] programs for Somali (SO), Swahili (SW), and Tagalog (TL). Publicly available resources from OPUS [124] and lexicons mined from Panlex [189] and Wiktionary are also used in the parallel corpus.

Experiments are evaluated on all partitions of MATERIAL (ANALYSIS, DEV, and EVAL), apart from Tagalog EVAL since GT judgments had not been released at the time. In order to compare the document-level relevance judgments, the sentence-level relevance scores from SECLR are aggregated to obtain document level scores.

The **SECLR** model and the SECLR model with rationale training (**SECLR-RT**) are compared to other cross-lingual embedding methods as follows: (1) **Bivec** [130] trained on parallel sentences; (2) **MUSE** [132] trained using bilingual dictionary from Wiktionary, and (3) **SID-SGNS** [131] trained on parallel sentences. We also compare our models to a pipeline of NMT [127] with monolingual IR (**NMT+IR**) and a pipeline of SMT, using Moses [190] and KenLM for LM [129], with monolingual IR (**SMT+IR**). We also compare our work with the **PSQ** model of [85] and the cross-lingual model **XLM-R** [78].

MAP is presented in Tables 63 and Table 64, which demonstrate that overall, SECLR-RT or SECLR are able to outperform the other baselines, where † indicates significance at the p = 0.01 level between SECLR-RT and the best baseline. To simulate data scarcity, we subsample the parallel corpus and present MAP scores of four models as a percentage of data sampled in Figure 24. Here, SECLR-RT is able to outperform the other baselines in most cases. In general, SECLR-RT is able to outperform SECLR, indicating that rationale training improves the robustness of the model.

Table 63. Experimental Results Comparing SECLR and SECLR-RT to Other Baselines by Document-Level MAP Scores for Text (T) and Speech (S) for Somali and Swahili.

	Somali						Swahili					
	Analysis		Dev		Eval		Analysis		Dev		Eval	
Method	T	S	T	S	T	S	T	S	T	S	T	S
Bivec	19.6	16.2	15.0	12.0	4.2	4.5	23.9	22.7	21.9	21.6	6.2	4.8
SID-SGNS	25.5	24.3	22.2	16.0	10.2	9.1	38.8	36.3	33.7	30.3	16.2	13.6
MUSE	9.9	9.9	10.3	16.5	1.9	2.0	27.8	24.5	27.3	28.8	9.5	8.1
NMT+IR	18.8	12.5	21.1	13.4	9.4	8.4	23.7	24.9	26.8	26.7	15.3	11.4
SMT+IR	17.4	11.2	19.1	16.8	9.1	8.3	25.5	28.6	27.1	25.2	15.4	13.3
PSQ	27.0	16.6	25.0	20.7	11.1	8.6	39.0	36.6	38.0	38.6	20.4	13.8
XLM-R	13.9	11.0	10.7	12.4	2.3	2.9	23.3	29.0	20.0	29.7	6.2	7.5
SECLR	27.8	24.4	23.0	17.4	7.7	7.4	43.8	37.9	**40.3**	38.1	16.0	13.1
SECLR-RT	**35.4†**	**28.4**	**29.5**	**22.0**	**13.1†**	**11.2†**	**48.3†**	**48.1†**	39.6	**45.4**	**22.7†**	**17.7†**

Table 64. Experimental Results Comparing SECLR and SECLR-RT to Other Baselines by Document-Level MAP Scores for Text (T) and Speech (S) for Tagalog.

	Analysis		Dev	
Method	T	S	T	S
Bivec	36.7	41.4	39.6	26.9
SID-SGNS	44.6	43.9	40.9	41.7
MUSE	27.4	26.5	26.0	16.5
NMT+IR	37.7	42.3	32.6	37.5
SMT+IR	44.4	52.7	39.3	35.3
PSQ	51.6	55.0	52.7	44.7
SECLR	46.7	45.0	49.3	33.9
SECLR-RT	**61.1**	**55.5**	**59.0**	**45.7**

SECLR-RT is also able to alleviate the hubness problem, where there exist a few vectors that are neighbors of many other vectors in the cross-lingual word embedding space. Table 65 shows S_N10 scores that measure the skewness of the distribution of *N*10, which indicates the size of the neighborhood around a vector in the sentence collection language embedding space. More details on this method can be found in [146].

Table 65. Experimental Results for SN_{10} scores of SECLR and SECLR-RT.

Model	Somali	Swahili	Tagalog
SECLR	29.36	54.98	43.29
SECLR-RT	**6.78**	**14.73**	**11.73**

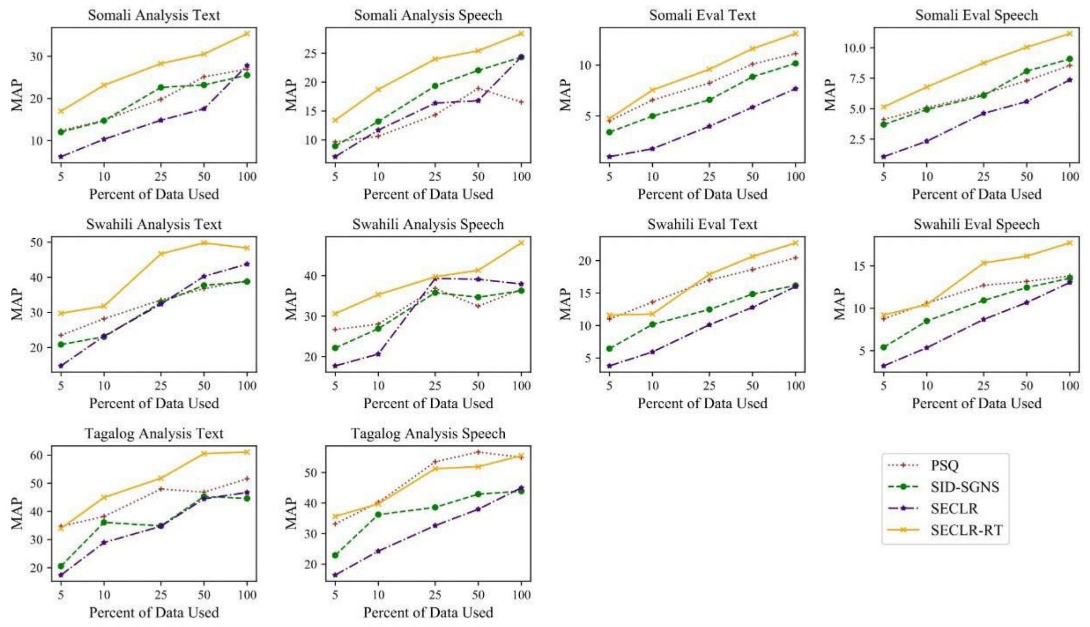


Figure 24. Ablation Study Results of SECLR and SECLR-RT Model Performances as a Function of Sub-Sampling Percentages.

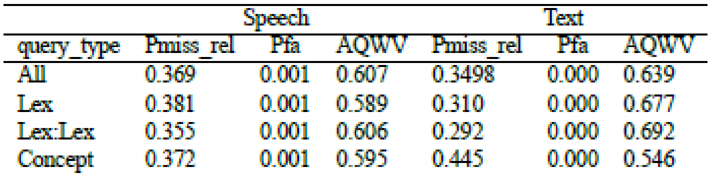
X-Coordinate uses the Log Scale.

4.6.7 E2E Evaluation Scores and Reclassification Rates for 3S

For Farsi (3S), the final system used the core unsupervised system along with the supervised approach. The embeddings used were Glove6B, Glove42B (300 dimensional), and RoBERTa for

English embeddings and fastText (300 dimensional) for foreign language embeddings. Although the speech segmentation system was available, there were insufficient data to improve performance, so speech segmentation was not used. Table 66 contains E2E evaluation results and Table 67 contains evaluation reclassification rates.

Table 66. E2E Results for Farsi EVAL.



	Speech			Text		
query_type	Pmiss_rel	Pfa	AQWV	Pmiss_rel	Pfa	AQWV
All	0.369	0.001	0.607	0.3498	0.000	0.639
Lex	0.381	0.001	0.589	0.310	0.000	0.677
Lex:Lex	0.355	0.001	0.606	0.292	0.000	0.692
Concept	0.372	0.001	0.595	0.445	0.000	0.546

Table 67. Reclassification Rates for Farsi EVAL.

	Speech			Text		
query_type	TD_accep t	FA_reject	H1	TD_accep t	FA_reject	H1
All	0.766	0.836	0.799	0.746	0.795	0.770
Lex	0.778	0.817	0.797	0.771	0.792	0.781
Lex:Lex	0.767	0.819	0.792	0.782	0.763	0.772
Concept	0.729	0.886	0.800	0.641	0.842	0.728

4.6.8 Use of Retranslation or APE

We also implemented re-translation or APE after the summaries were generated to re-insert query terms into the summary when we found evidence that the query term should have appeared in the translation. We ran these experiments on Farsi after the Farsi EVAL to investigate the usefulness of retranslation. We also implemented this component for Kazakh and Georgian.

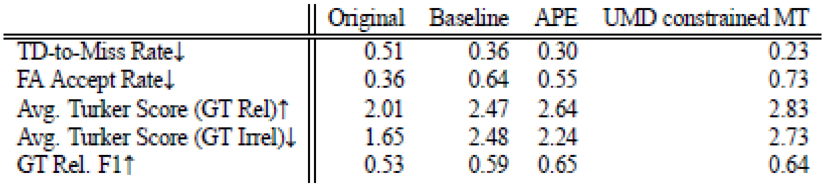
Our initial retranslation experiments were run after the Farsi evaluation period in OP2 on a sample of 1,000 Farsi EVAL query-document pairs, equally split between text and audio and between irrelevant and relevant documents. These query-document pairs were selected such that the summaries we produced did not contain the query word. We experimented with several retranslation strategies (UMD constrained MT, APE, and a baseline string replacement approach based on finding the query string in UMDNMT's *n*-best translation lattices), as well as several strategies for selecting which sentence in the summary should be retranslated (making use of CLIR's PSQ evidence as well as the *n*-best translation lattices from UMDNMT). However, since sentence selection strategies often agreed and did not impact results significantly, they were omitted from the following results. Both APE and constrained MT increased FA-accept significantly even as they decreased TD-to-miss rate as compared to results without retranslation. This indicates that further work was needed to select which summaries were retranslated.

In Table 68 and Table 69, we evaluate several metrics: the percentage of relevant summaries marked irrelevant by AMT workers (**TD-to-Miss Rate**), the percentage of irrelevant summaries marked relevant (**FA-Accept Rate**), and the average AMT worker score of ground-truth relevant (GT Rel) and irrelevant (GT Irrel) summaries on a scale of 0 to 4, where 0 is definitely irrelevant, 2 is unsure, and 4 is definitely relevant. We also include the F1 score of the positive (relevant summary) class.

Initial experiments in Kazakh were done on a sample of 1,000 DEV query-document pairs for which our summaries did not contain query words. We also additionally experimented with the constrained MT system developed by UMD. Results in Table 69 similarly show that any decrease in TD-to-Miss was accompanied by an increase in recall.

Further experiments in Kazakh were done using the document selection strategy described in Section 2.7.4 where we use a logistic regressor to select GT relevant documents and then use different ways of selecting the threshold. A sample of 2,000 DEV summaries was built from collections of documents selected via different strategies so that we could compare their impact. Different thresholds were simulated afterwards and are shown in Table 70 and Table 71.

Table 68. Experimental Results Demonstrating Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Farsi EVAL Sample.



	Original	Baseline	APE	UMD constrained MT
TD-to-Miss Rate↓	0.51	0.36	0.30	0.23
FA Accept Rate↓	0.36	0.64	0.55	0.73
Avg. Turker Score (GT Rel)↑	2.01	2.47	2.64	2.83
Avg. Turker Score (GT Irrel)↓	1.65	2.48	2.24	2.73
GT Rel. F1↑	0.53	0.59	0.65	0.64

Table 69. Results Showing Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Kazakh DEV Sample without Document Selection Strategies.

	Original	APE	Edi constrained MT	UMD constrained MT
TD-to-miss	***0.363***	**0.190**	**0.130**	**0.222**
FA-accept	0.344	0.617	0.802	0.620
Turkers (Rel)	2.47	2.93	3.18	2.90
Turkers (Irrel)	1.69	2.48	3.00	2.48
Precision	0.442	0.384	0.339	0.373
Recall	***0.637***	**0.810**	**0.870**	**0.778**
F1	0.522	0.521	0.488	0.504

Based on these results, we selected an F1-based cutoff threshold as our document selection strategy and used APE for retranslation since that had the highest F1.

Table 70. Results Showing Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Kazakh DEV Sample Using CLIR F1 Threshold.

	Original	APE	Edi constrained MT	UMD constrained MT
TD-to-miss	0.391	0.250	0.191	0.293
FA-accept	0.365	0.531	0.631	0.540
Turkers (Rel)	2.37	2.83	3.08	2.69
Turkers (Irrel)	1.72	2.23	2.53	2.24
Precision	0.344	0.311	0.290	0.294
Recall	0.609	0.750	0.809	0.707
F1	0.440	**0.439**	0.427	0.416

Table 71. Results Showing Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Kazakh DEV Sample Using F1 Threshold.

Original		APE	Edi constrained MT	UMD constrained MT
TD-to-miss	0.391	0.272	0.229	0.318
FA-accept	0.365	0.450	0.477	0.439
Turkers (Rel)	2.37	2.77	2.93	2.61
Turkers (Irrel)	1.72	1.98	2.06	1.94
Precision	0.344	0.340	0.340	0.331
Recall	0.609	0.728	0.771	0.682
F1	0.440	***0.464***	0.472	0.446

We did similar experiments with Georgian DEV+ANALYSIS and achieved the results shown in Tables 72, 73, and 74. For Georgian, we omitted UMD constrained MT but added APE on the Edinburgh constrained MT output. Due to these results, we chose EDIN constrained MT with F1 threshold to maximize F1-score.

Table 72. Results Showing Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Georgian DEV+ANALYSIS Sample Using CLIR F1 Threshold.

Original		APE	Edi constrained MT	APE on Edi constrained MT
TD-to-miss	0.440	0.397	0.272	0.289
FA-accept	0.420	0.432	0.584	0.543
Turkers (Rel)	2.34	2.36	2.80	2.75
Turkers (Irrel)	1.88	1.92	2.36	2.26
Precision	0.245	0.252	0.232	0.241
Recall	0.560	0.603	0.728	0.711
F1	0.341	0.355	**0.351**	0.359

Table 73. Results Showing Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Georgian DEV+ANALYSIS Sample Using ROC Threshold.

Original		APE	Edi constrained MT	APE on Edi constrained MT
TD-to-miss	0.440	0.399	0.209	0.235
FA-accept	0.420	0.435	0.669	0.620
Turkers (Rel)	2.34	2.36	2.96	2.89
Turkers (Irrel)	1.88	1.91	2.58	2.47
Precision	0.245	0.251	0.223	0.230
Recall	0.560	0.601	0.791	0.765
F1	0.341	0.354	**0.347**	0.354

Table 74. Results Showing Impact of Retranslation on TD-to-Miss Rate, FA Accept Rate, and F1 Score for Georgian DEV+ANALYSIS Sample Using F1 Threshold.

Original		APE	Edi constrained MT	APE on Edi constrained MT
TD-to-miss	0.440	0.403	0.287	0.305
FA-accept	0.420	0.427	0.496	0.481
Turkers (Rel)	2.34	2.37	2.75	2.71
Turkers (Irrel)	1.88	1.90	2.10	2.07
Precision	0.245	0.253	0.258	0.259
Recall	0.560	0.597	0.713	0.695
F1	0.341	0.355	**0.379**	0.377

4.6.8.1 APE

Below we have experimental results specific to the development of the APE model.

While developing the APE model, we considered the use of the Multi-source Transformer instantiation (MST) model [191] based on the AT APE model, and the non-autoregressive LevT [192]. For decoding, we experimented with the AT and LevT decoders.

For encoding, we experimented with several techniques to incorporate constraints as input to the APE encoder. There are two main ways constraints were incorporated into the source sequence:

- **Append**: Adding the target language constraint terms after their source language constraint terms within the source sequence.
- **Replace**: Replacing the source language constraint terms with their corresponding target language constraint terms.

In both cases, a source factor is associated with each token in the input sequence to indicate whether it is a source or target side terminology constraint, or if it is an unconstrained source token.

Methods of incorporating these modified input sequences with the MST and LevT models are described in more detail in [193]. In total, we experimented with three variants of MST:

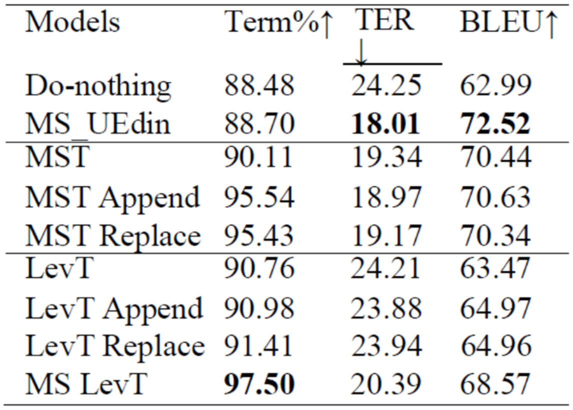
- unconstrained baseline (MST)
- constrained version using append (MST append)
- constrained version using replace (MST replace)

And four variants of LevT:

1. unconstrained model (LevT)
2. constrained variant incorporating constraint using append (LevT append)
3. constrained variant incorporating constraint using replace (LevT replace)
4. variant where the source input and MT input to correct are passed into separate encoders where decoder is initialized with target sequence of terminology constraints (MS LevT).

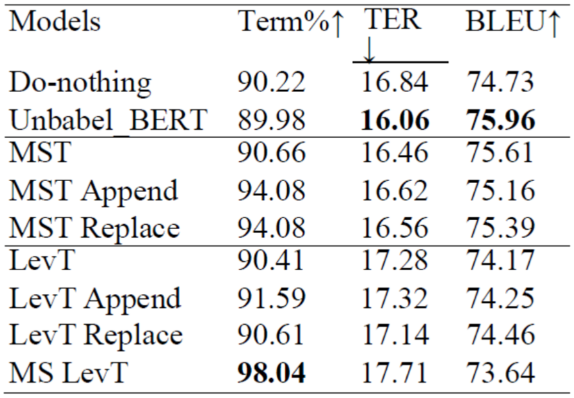
We test on both PBMT and NMT English to German WMT APE tasks [140]. Results for PBMT 2018 are shown in Table 75 and results for NMT 2019 are shown in Table 76. Generally, models are able to increase Term% with variants having different impacts on BLEU and TER.

Table 75. Experimental Results of APE Model Variants on PBMT 2018.



Models	Term%↑	TER ↓	BLEU↑
Do-nothing	88.48	24.25	62.99
MS_UEdin	88.70	**18.01**	**72.52**
MST	90.11	19.34	70.44
MST Append	95.54	18.97	70.63
MST Replace	95.43	19.17	70.34
LevT	90.76	24.21	63.47
LevT Append	90.98	23.88	64.97
LevT Replace	91.41	23.94	64.96
MS LevT	**97.50**	20.39	68.57

Table 76. Experimental Results of APE Model Variants on NMT 2019.



Models	Term%↑	TER ↓	BLEU↑
Do-nothing	90.22	16.84	74.73
Unbabel_BERT	89.98	**16.06**	**75.96**
MST	90.66	16.46	75.61
MST Append	94.08	16.62	75.16
MST Replace	94.08	16.56	75.39
LevT	90.41	17.28	74.17
LevT Append	91.59	17.32	74.25
LevT Replace	90.61	17.14	74.46
MS LevT	**98.04**	17.71	73.64

Our results also indicate that constrained APE improves translation quality and term preservation on both unconstrained and constrained MT. While constrained APE and unconstrained APE have similar results on systemic errors in MT output, constrained APE is able to preserve term constraints. For these experiments, MST is used for the unconstrained APE and MST append is used for constrained APE. A constrained and unconstrained AT MT model is used for unconstrained and constrained MT respectively. Results for different combinations of APE and MT systems are shown in Table 77.

Table 77. Experimental Results of Different Combinations of MT and APE Systems.
Constrained MT and APE are Indicated as cMT and cAPE Respectively.

Pipeline	Term%↑	TER ↓	BLEU↑
MT	45.33	70.78	15.28
cMT	86.33	70.24	15.47
MT → APE	55.35	59.56	22.87
cMT → APE	77.22	59.78	23.03
MT → cAPE	80.18	58.70	23.95
cMT → cAPE	88.38	59.77	23.08

After analyzing translation behavior, we found that rare or unusual terminology constraints that are in conflict with the decoder LM will give higher probabilities to frequently occurring term translations. In order to allow for *systemic copying* - where the model enforces term constraints even when they strongly disagree with the decoder LM - we experimented with data augmentation. We created novel instances in our training set by replacing the target language term with either a synonym or antonym and also replacing its occurrence in the post edited target translation. Table 78 shows the results with data augmentation, where we experimented with using the augmented corpus for both the pre-training and the fine-tuning process.

Table 78. Results for APE with Data Augmentation on the Official APE Data, and the Augmented Dataset of Synonyms and Antonyms Generated from Wiktionary.

	WMT'19 APE			Augmentation		
	Term% ↑	TER↓	BLEU ↑	Term% ↑	TER ↓	BLEU ↑
Do-nothing	90.22	16.84	74.73	1.66	24.77	62.56
MST Append	94.08	16.62	75.16	7.47	24.92	61.80
MST Append + pretrain	94.08	16.46	75.25	18.67	23.70	64.38
MST Append + pretrain + ft	93.85	**16.29**	**75.38**	43.15	**21.85**	**67.41**
MS LevT	98.04	17.71	73.64	43.57	33.07	54.33
MS LevT + pretrain	**99.09**	17.18	74.22	52.70	29.79	60.24
MS LevT + pretrain + ft	98.41	17.00	74.66	**63.07**	29.66	60.47

4.6.9 E2E Evaluation Scores and Reclassification Rates for 3C and 3B

For Kazakh (3C) and Georgian (3B) in OP2, the final systems used the unsupervised approach, supervised approach, and retranslation. The embeddings used were the same as the final Farsi system. For Kazakh, we utilized APE as the retranslation strategy while for Georgian, we chose EDIN constrained MT. Table 79 contains Kazakh E2E evaluation results and Table 80 contains Kazakh reclassification rates. Table 81 contains Georgian E2E evaluation results and Table 82 contains Georgian reclassification rates.

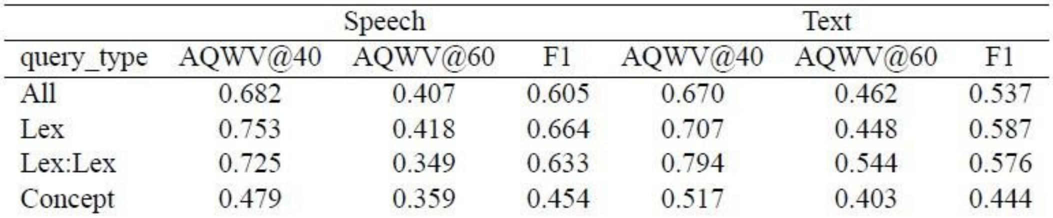
Table 79. E2E Results for Kazakh EVAL.

	Speech			Text		
query_type	AQWV@40	AQWV@60	F1	AQWV@40	AQWV@60	F1
All	0.721	-0.135	0.453	0.800	0.432	0.455
Lex	0.693	-0.287	0.460	0.751	0.326	0.468
Lex:Lex	0.749	-0.440	0.428	0.792	0.378	0.462
Concept	0.706	0.263	0.477	0.852	0.575	0.437

Table 80. Reclassification Rates for Kazakh EVAL.

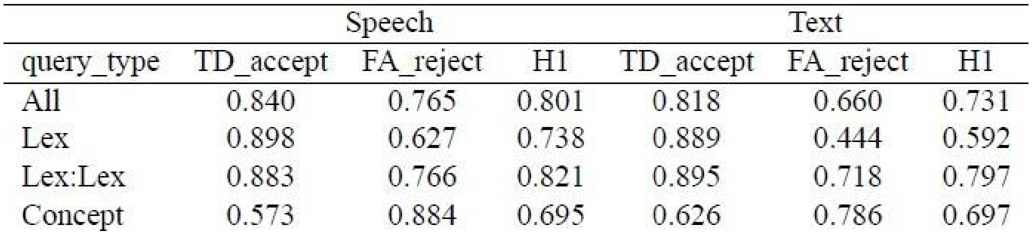
	Speech			Text		
query_type	TD_accept	FA_reject	H1	TD_accept	FA_reject	H1
All	0.909	0.671	0.772	0.865	0.728	0.791
Lex	0.935	0.565	0.704	0.880	0.634	0.737
Lex:Lex	0.943	0.594	0.728	0.895	0.705	0.789
Concept	0.792	0.82	0.810	0.813	0.828	0.820

Table 81. E2E Results for Georgian EVAL.



	Speech			Text		
query_type	AQWV@40	AQWV@60	F1	AQWV@40	AQWV@60	F1
All	0.682	0.407	0.605	0.670	0.462	0.537
Lex	0.753	0.418	0.664	0.707	0.448	0.587
Lex:Lex	0.725	0.349	0.633	0.794	0.544	0.576
Concept	0.479	0.359	0.454	0.517	0.403	0.444

Table 82. Reclassification Rates for Georgian EVAL.



	Speech			Text		
query_type	TD_accept	FA_reject	H1	TD_accept	FA_reject	H1
All	0.840	0.765	0.801	0.818	0.660	0.731
Lex	0.898	0.627	0.738	0.889	0.444	0.592
Lex:Lex	0.883	0.766	0.821	0.895	0.718	0.797
Concept	0.573	0.884	0.695	0.626	0.786	0.697

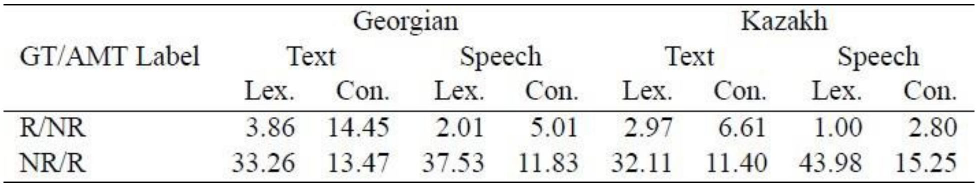
We also conducted additional performance analysis and analysis of inter-annotator agreement on Kazakh and Georgian evaluation results.

4.6.9.1 Performance Analysis

We used AMT summary evaluation results to conduct performance analysis on Kazakh and Georgian languages. More specifically, we used AMT judge triplets to assign relevance labels to query-document pairs using majority rule based on the AMT judge labels of “relevant” and “not relevant” regardless of qualifier. For example, if two out of the three judge labels were either “probably” or “definitely” not relevant, we would label the query-document pair irrelevant. In our analysis, we focused on query-document pairs where: (1) the document GT was relevant and the AMT label is irrelevant (**R/NR**), and (2) the document GT was irrelevant and the AMT label is relevant (NR/R). We did the analysis across both lexical (Lex.) and conceptual (Con.) queries on each of the two document source modalities: text (NT, BT) and speech (CS, NB, TB). Table 83 shows the results across both languages, where the percentage of query-document pairs labeled as

GT relevant and AMT irrelevant are labeled (R/NR) and those labeled GT irrelevant and AMT relevant are labeled (NR/R). Each number represents the percentage of query- document pairs for the given document source type. For example, out of all the query- document pairs in Georgian where the document was speech based and the query was lexical 37.53% of the pairs had non-relevant GT documents assigned as AMT relevant.

Table 83. AMT Results - Georgian and Kazakh



GT/AMT Label	Georgian				Kazakh			
	Text		Speech		Text		Speech	
	Lex.	Con.	Lex.	Con.	Lex.	Con.	Lex.	Con.
R/NR	3.86	14.45	2.01	5.01	2.97	6.61	1.00	2.80
NR/R	33.26	13.47	37.53	11.83	32.11	11.40	43.98	15.25

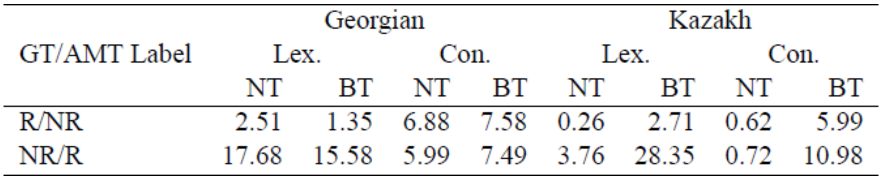
Across both languages and document source modalities (text and speech), we observe a higher percentage of GT relevant documents labeled as irrelevant by AMT judges (R/NR) with conceptual queries as compared to lexical queries. On the other hand, for GT irrelevant documents labeled as relevant by AMT judges (NR/R), we observe the opposite; in this case, the percentage is higher for lexical queries.

Across most of the document source and query type tuples (e.g., Speech-Lex.), we see that NR/R document labels are more dominant than R/NR, except for the case of the Georgian Text-Con. where R/NR labels are slightly higher. Across both languages and document label types (i.e., R/NR and NR/R), we observe best summarization performance on instances where the document source type is speech and the query is lexical, with 2.01% and 1.00% respectively.

Table 84 provides the breakdown of the above percentages for documents whose source is text. We provide the breakdown across the two text types: News text (NT) and blog text (BT).

Relevant BT documents are labeled irrelevant by AMT judgements (R/NR) more frequently than NT documents, with the exception of lexical queries on Georgian language documents. A similar pattern can be observed across NR/R labeled documents.

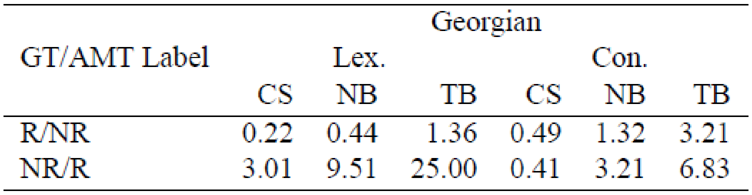
Table 84. AMT Results - Georgian and Kazakh (Text Only).



GT/AMT Label	Georgian				Kazakh			
	Lex.		Con.		Lex.		Con.	
	NT	BT	NT	BT	NT	BT	NT	BT
R/NR	2.51	1.35	6.88	7.58	0.26	2.71	0.62	5.99
NR/R	17.68	15.58	5.99	7.49	3.76	28.35	0.72	10.98

We perform similar breakdown for documents whose source is speech. In Tables 85 and 86 for each language we provide the breakdown across the three speech types: CS, NB, and TB. The percentage of query-document pairs labeled as GT relevant and AMT irrelevant are labeled (R/NR) and those labeled GT irrelevant and AMT relevant are labeled (NR/R).

Table 85. AMT Results for Georgian (Speech Only).



	Georgian					
GT/AMT Label	Lex.			Con.		
	CS	NB	TB	CS	NB	TB
R/NR	0.22	0.44	1.36	0.49	1.32	3.21
NR/R	3.01	9.51	25.00	0.41	3.21	6.83

Table 86. AMT Results for Kazakh (Speech Only).

	Kazakh					
GT/AMT Label	Lex.			Con.		
	CS	NB	TB	CS	NB	TB
R/NR	0.00	0.36	0.64	0.00	0.80	2.00
NR/R	0.45	14.53	29.00	0.00	5.08	10.17

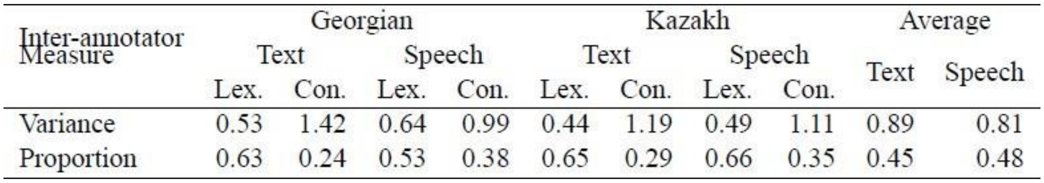
Across both languages, document label (R/NR and NR/R) and query types for most of the documents originate from TB. The fewest originate from CS. In case of the Kazakh language, none of the documents labeled R/NR originate from CS.

4.6.9.2 Inter-annotator Agreement Analysis

We also analyzed agreements across AMT annotators. Annotator agreements were analyzed using two measures: inter-annotator variance, and proportion of inter-annotator agreements. These results are shown in Table 87 across the two languages, document source and query types.

In the last two columns of Table 87, we present the average inter-agreement variance and agreement proportion for text and speech document source modalities across both languages and query types. From the average values, we observe that inter-annotator agreement for text is slightly lower than speech document types. This is also related to higher average inter-annotator variance on text documents versus speech.

Table 87. Inter-Annotator Agreement Analysis: Inter-Annotator Agreement Variance and Agreement Proportion across Georgian and Kazakh Languages over Different Document Source and Query Types.



Inter-annotator Measure	Georgian				Kazakh				Average	
	Text		Speech		Text		Speech		Text	Speech
	Lex.	Con.	Lex.	Con.	Lex.	Con.	Lex.	Con.		
Variance	0.53	1.42	0.64	0.99	0.44	1.19	0.49	1.11	0.89	0.81
Proportion	0.63	0.24	0.53	0.38	0.65	0.29	0.66	0.35	0.45	0.48

To understand whether most documents with high variance are indeed text documents, we performed a more fine-grained analysis where we sorted documents in both languages in descending order based on their variance values. By focusing on the top 20% highest-variance documents, we discovered that 85.2% of the Georgian and 77.6% of Kazakh documents are text based.

From Table 87, we also observe that the level of agreement across both languages and document source modalities is lower for conceptual versus lexical queries. Inter-annotator variance across both languages and document source modalities is also higher for conceptual queries versus lexical. To understand whether most queries with high inter-annotator agreement variance are indeed conceptual, we sorted queries in both languages in descending order based on their variance values. By once again focusing on the top 20% of highest variance queries, we discovered 75.6% of the Georgian and 80.9% of Kazakh queries are conceptual. In Table 88, we provide examples of conceptual queries with large variance values; although this is a small sample size, we note that they focus on specific global events with which an average annotator may or may not be familiar.

Table 88. Inter-Annotator Agreement Analysis: Example Conceptual Queries with Large Inter-Annotator Agreement Variance.

Query	Inter-annotator Variance
closure of the Eiffel Tower+	3.16
threats from the Pentagon+	3.12
Pushkin Square renovations+	2.52
offers by Beeline+	2.44
military activities in Bagram+	2.44
diplomatic recognition by Nicaragua+	2.37

In Table 89 we provide analysis of inter-annotator agreements across GT irrelevant (**NR**) and relevant (**R**) documents.

Inter-annotator agreement is lower for GT irrelevant documents (**NR**) compared to GT relevant documents (**R**). Given that AMT judges only annotate CLIR relevant documents, this suggests that annotators are more likely to disagree on summaries of the GT irrelevant documents that CLIR returned as relevant.

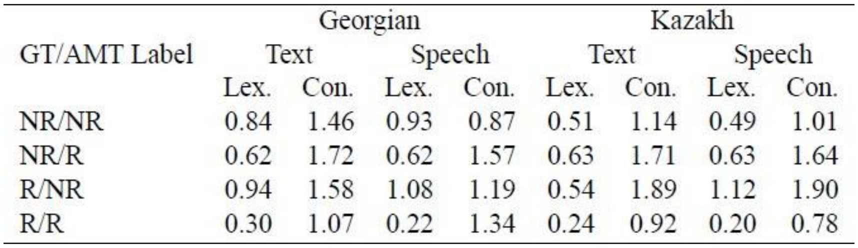
Table 89. Inter-Annotator Agreement Analysis: Inter-Annotator Agreement Variance and Agreement Proportion across GT Relevant (R) and Irrelevant (NR) Documents.

Inter-annotator Measure	GT Label	Georgian				Kazakh			
		Text		Speech		Text		Speech	
		Lex.	Con.	Lex.	Con.	Lex.	Con.	Lex.	Con.
Variance	NR	0.70	1.52	0.77	0.95	0.58	1.24	0.58	1.12
	R	0.35	1.26	0.28	1.27	0.26	1.10	0.24	1.01
Proportion	NR	0.53	0.22	0.45	0.39	0.60	0.24	0.62	0.34
	R	0.73	0.28	0.75	0.33	0.72	0.40	0.77	0.43

In Table 90, we provide a breakdown of inter-annotator agreement variance across the four different combinations of GT and AMT relevance labels: (1) GT and AMT irrelevant (NR/NR), (2) GT irrelevant and AMT relevant (Table 87), (3) GT relevant and AMT irrelevant (Table 87), and (4) GT and AMT relevant (Table 87).

Table 90. Inter-Annotator Agreement Analysis: Inter-Annotator Agreement Variance for Georgian and Kazakh by Document Source and Query Type.

Agreement variance is presented over all possible combinations of GT and AMT relevance labels.



	Georgian				Kazakh			
GT/AMT Label	Text		Speech		Text		Speech	
	Lex.	Con.	Lex.	Con.	Lex.	Con.	Lex.	Con.
NR/NR	0.84	1.46	0.93	0.87	0.51	1.14	0.49	1.01
NR/R	0.62	1.72	0.62	1.57	0.63	1.71	0.63	1.64
R/NR	0.94	1.58	1.08	1.19	0.54	1.89	1.12	1.90
R/R	0.30	1.07	0.22	1.34	0.24	0.92	0.20	0.78

Across each document source and query type, we see the highest inter-annotator agreement variance for NR/R and R/NR labels. Given that these are averages of the inter-annotator agreement variances across all documents in that label category, if the inter-annotator agreement variance is high for the given document source and query type, the AMT label should not be trusted.

4.6.10 Other Experimental Approaches

Experimental results for the new approaches that did not become part of the submitted system are included below.

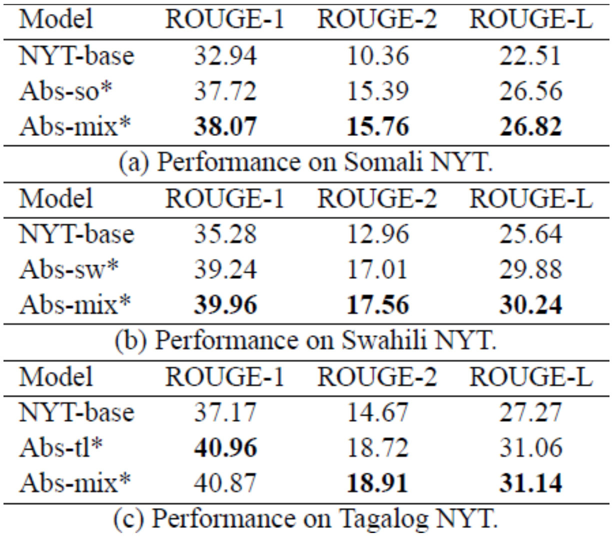
4.6.10.1 Abstractive Cross-lingual Summarization

The data used to train the abstractive summarization model is the *NYT* summarization corpus [91], which consists of articles and their human written abstractive summaries. Articles are translated into three LRLs (Somali, Swahili and Tagalog) and then translated back into noisy English in order to form noisy article and clean English reference summary pairs. More detail and experimental results can be found in [142].

The baseline system is pre-trained on the unmodified NYT corpus. The three noisy English corpora are each used to train the baseline system for another eight epochs, resulting in two language-specific abstractors. A fourth abstractor is trained using articles randomly selected from the three noisy corpora. Table 91 shows the performance of the abstractors on the Somali, Swahili and Tagalog NYT test sets as measured by various variants of the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metric.

Table 91. Experimental Results of Abstractive Cross-Lingual Summarization Systems on the NYT Test Sets.

*Abs-so, -sw and -tl are the Somali, Swahili, and Tagalog systems, respectively. * indicates significant improvement over NYT-base (p < 0.01).*



Model	ROUGE-1	ROUGE-2	ROUGE-L
NYT-base	32.94	10.36	22.51
Abs-so*	37.72	15.39	26.56
Abs-mix*	**38.07**	**15.76**	**26.82**

(a) Performance on Somali NYT.

Model	ROUGE-1	ROUGE-2	ROUGE-L
NYT-base	35.28	12.96	25.64
Abs-sw*	39.24	17.01	29.88
Abs-mix*	**39.96**	**17.56**	**30.24**

(b) Performance on Swahili NYT.

Model	ROUGE-1	ROUGE-2	ROUGE-L
NYT-base	37.17	14.67	27.27
Abs-tl*	**40.96**	18.72	31.06
Abs-mix*	40.87	**18.91**	**31.14**

(c) Performance on Tagalog NYT.

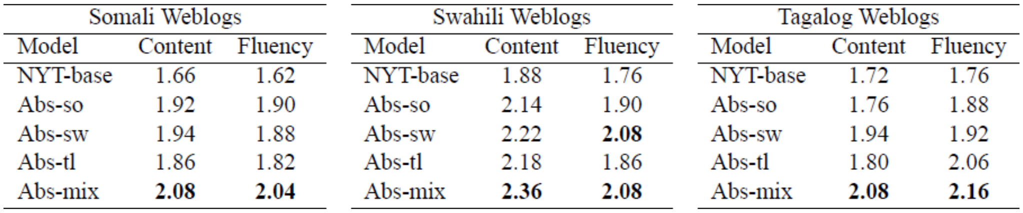
The mixed model, which uses articles from all three corpora, achieved the best scores. The perplexity of the abstractors' output was used as a proxy for fluency and measured; the results are shown in Table 92. All models produce more fluent English across source languages than the base model. Similar human evaluations were done on Somali, Swahili and Tagalog weblogs to compare the content and fluency of the summaries and saw that models were able to improve in fluency over the base model. See Table 93 for these results.

Table 92. LM Perplexity of Summaries Generated by Abstractive Cross-Lingual Summarization Systems on Noisy Somali, Swahili, and Tagalog NYT.

	Perplexity		
Model	Somali NYT	Swahili NYT	Tagalog NYT
NYT-base	4986	4428	4707
Abs-so	**3357**	3429	3528
Abs-sw	3384	**3247**	**3312**
Abs-tl	3501	3476	3457
Abs-mix	3464	3285	3402

Table 93. Experimental Results with Human Annotators on Output from Abstractive Cross-lingual Summarization Systems.

Average human-rated content and fluency scores on Somali, Swahili and Tagalog weblog entries.



Somali Weblogs			Swahili Weblogs			Tagalog Weblogs		
Model	Content	Fluency	Model	Content	Fluency	Model	Content	Fluency
NYT-base	1.66	1.62	NYT-base	1.88	1.76	NYT-base	1.72	1.76
Abs-so	1.92	1.90	Abs-so	2.14	1.90	Abs-so	1.76	1.88
Abs-sw	1.94	1.88	Abs-sw	2.22	**2.08**	Abs-sw	1.94	1.92
Abs-tl	1.86	1.82	Abs-tl	2.18	1.86	Abs-tl	1.80	2.06
Abs-mix	**2.08**	**2.04**	Abs-mix	**2.36**	**2.08**	Abs-mix	**2.08**	**2.16**

The system was subsequently evaluated on Arabic using the DUC 2004 Task 4 test set [194]. The performance of the abstractors are shown in Table 94 (evaluated on the DUC corpus with translations provided by systems from the Information Sciences Institute (ISI) at the University of Southern California), where * indicates significant improvement over NYT-base ($p < 0.01$); † indicates significant difference between systems ($p < 0.05$). The results show that there is a significant improvement in ROUGE [195] as compared to past task systems and show that the abstractors are able to generalize and improve fluency of summaries in a previously unseen language.

Table 94. Experimental Results of Abstractive Cross-Lingual Summarization Systems Evaluated on DUC 2004 with ISI Translations.

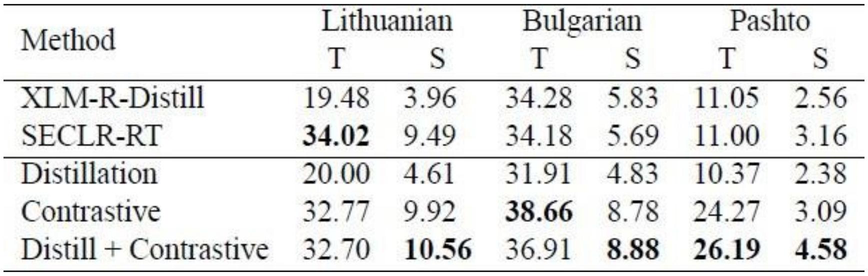
Model	ROUGE-1	ROUGE-2	ROUGE-L
NYT-base	26.56	5.86	15.76
Abs-so*	28.64	6.66	19.62
Abs-sw*†	28.08	6.39	18.36
Abs-tl*†	**29.43**	**7.02**	**19.89**
Abs-mix	28.79	6.74	19.79

4.6.10.2 Conceptual Query Processing

In order to generate training data, parallel sentences were used, and the query was formed by using dependency parsing to randomly select noun phrases from the English sentence. Training data were generated using parallel data for source languages and English pairs available in OPUS [124], a large collection of parallel corpora. Results of the model are evaluated on the DEV collection of MATERIAL.

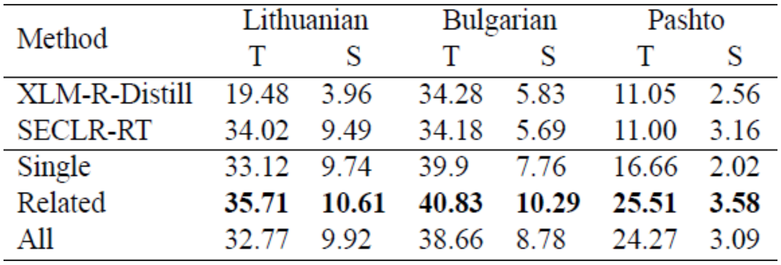
We experiment with several training objectives to learn semantic similarity and minimize the distance between similar embeddings: distillation training, contrastive training, and distillation training as a pre-training step *before* contrastive training. Results training monolingual models are shown in Table 95 - where we report MAP on the DEV collection. T and S stands for text and speech, respectively - while the results shown for multilingual training - where the model is trained on concatenations of datasets of related languages - are shown in Table96, where we report MAP on DEV. We present our results using contrastive learning with only the corresponding language (Single), languages related to the source language (Related), and the concatenation of all the languages (All). SECLR-RT is discussed in Section 3.7.3 and XLM-R-Distill, introduced in [196], is the fine-tuned XLM-RoBERTa model trained on distillation. More detail about this approach and the results can be found in [197]. From these tables, we note our approach using distillation and contrastive learning works best for Pashto; the results are mixed for other languages, with speech performance increasing in Bulgarian and Lithuanian, and text performance improving on Bulgarian (with the contrastive approach alone) but not Lithuanian. However, augmenting the data with related languages for multilingual training performs best across the board for all conditions.

Table 95. Experimental Training MAP Results for the Training Objective Experiment on the Conceptual Query Relevance Model.



Method	Lithuanian		Bulgarian		Pashto	
	T	S	T	S	T	S
XLM-R-Distill	19.48	3.96	34.28	5.83	11.05	2.56
SECLR-RT	**34.02**	9.49	34.18	5.69	11.00	3.16
Distillation	20.00	4.61	31.91	4.83	10.37	2.38
Contrastive	32.77	9.92	**38.66**	8.78	24.27	3.09
Distill + Contrastive	32.70	**10.56**	36.91	**8.88**	**26.19**	**4.58**

Table 96. Experimental Results for Multilingual Training of our Conceptual Query Relevance Model.



Method	Lithuanian		Bulgarian		Pashto	
	T	S	T	S	T	S
XLM-R-Distill	19.48	3.96	34.28	5.83	11.05	2.56
SECLR-RT	34.02	9.49	34.18	5.69	11.00	3.16
Single	33.12	9.74	39.9	7.76	16.66	2.02
Related	**35.71**	**10.61**	**40.83**	**10.29**	**25.51**	**3.58**
All	32.77	9.92	38.66	8.78	24.27	3.09

4.6.11 Technical Insights

Here we draw larger lessons from our work on summarization in MATERIAL.

4.6.11.1 Importance of Effective Methods for Synthetic Dataset Creation

In the base period of MATERIAL, we had little information to inform us about whether the approach we were developing was effective; there were no training or test data available whatsoever. While we carried out multiple AMT tests in this phase, their scope was necessarily limited. In most query-focused summarization scenarios, this is also the case: it is rare to have a large volume of data for a query-focused task. In fact, in DUC evaluations from 2004 onward, hand annotated query-focused datasets are usually limited to 100 or fewer query/document/ summary instances and query-focused multilingual datasets are non-existent. Thus, our work developing methods to create synthetic data for training could benefit many research programs in this area. We began with methods to create cross-lingual query/sentence relevance pairs using parallel data and creating queries from the English side of an English/foreign sentence pair simply by extracting words from the sentence. This allowed us, for the first time, to generate supervised methods for the lexical query task. Once we discovered the effectiveness of this method, we began to extend it to other tasks but there is still work to be done creating synthetic data for conceptual queries and for query-focused abstractive summarization methods.

4.6.11.2 Rationale Training Boosts Results

In our work on supervised summarization, we saw that adding rationales from phrase-based SMT enhanced the system effectiveness in selecting sentences relevant to a given query. Exploring other forms of rationale training for other aspects of the query-focused tasks (e.g., conceptual queries, abstractive methods) could yield additional improvements in performance.

4.6.11.3 Fluency Matters for Summary Effectiveness

We saw early on that when summaries were not fluent, AMT workers were often unable to correctly identify relevant documents even when the summary contained the query word; they either missed the query word or were unsure whether it had the appropriate meaning given the disfluent context. We experimented with a variety of approaches to improve fluency problems, which were especially prevalent in spoken language dialogues generated by both ASR and MT. We experimented with speech segmentation and saw some gains from this approach, but its applicability is limited to cases where there is adequate subtitle data to train systems. We also experimented with abstractive methods of summarization, hypothesizing that it may be easier for summarization to rewrite small segments of the input translations since the entire document was not needed. Our work on abstractive summarization did not make it into the final system in large part due to lack of training data, though we did develop methods for generating synthetic data for this task near the end of the program. As a result, there is ample room to explore how summary fluency can be improved. More research on summarizing informal language such as conversations to produce a meaningful, fluent summary is needed. Methods for producing fluent summarized text may be within closer reach, but still should be explored.

4.6.11.4 MT Quality has a Dramatic Impact on Summarization Effectiveness

In the BP and OP1, we noticed that our main problem was in TD-to-Miss. That is, we were unable to effectively demonstrate true document relevance to end users through a summary. We hypothesized this was because our summary methods couldn't locate the sentences that contained the query term or its synonyms. Towards the end of OP1, we discovered that our methods actually *were* finding the relevant sentences, but the query word simply did not appear in the translations. This led to our focus on automatic post editing and constrained decoding in MT. Further exploration of how MT and summarization could be jointly learned and performed could yield further improvements in this area.

4.6.11.5 Relaxing Metrics and Constraints on the Task

The summarization task as defined for MATERIAL was governed by the use of very specific query-types and constraints on the definition of what counted as correct. Summarization was also very driven by the metrics used. This led to a focus on methods that used various forms of extraction, pulling either snippets or full sentences from the input document. A less rigid approach to the task could enable more creativity on the part of performers in coming up with solutions that truly support human end users in finding the LRL documents they need. Allowing multiple ways of expressing information needs in tandem with a new evaluation design could lead to larger advances in summarization efficacy for human end users. We think abstractive summarization could play a larger role in presenting summaries that an end user can understand, but the current evaluation framework did not reward performers in this direction. We recommend experimenting with different task settings, evaluation design, and metrics to encourage further creativity in solutions and a more varied approaches to satisfying end user needs.

5.0 TRANSITION

5.1 SCRIPTS Full System Demo

This is an interactive cross-lingual query and summarization demo, allowing any query in the IARPA format as input and operating on EVAL data packs.

5.2 Morphagram

This is an unsupervised morphological segmentation framework that is language independent. The input to the framework is an unlabeled list of words and the output is the morphological segmentation of the input words and a grammar that parses unseen words. Adding linguistic knowledge (in terms of prefixes and suffixes) is an option.

The framework is also available at: https://github.com/rnd2110/MorphAGram.

5.3 SCRIPTS Morphological Analyzer

This is described as a morphological-analysis system that does morphological segmentation and morphological tagging for all MATERIAL languages. The analyzer can be executed through a runnable Java Archive (JAR) or as a Docker image, in either a standalone mode or a client-server one. For each word in a given context, the analyzer produces the following information:

- Word POS (universal POS tagset)
- Word segmentation (prefixes, stem and suffixes)
- Word Tense (past/present)
- Word Number (singular/plural)

The latest neural version of the analyzer is also available at: https://github.com/rnd2110/unsupervised-cross-lingual-POS-tagging. The tagger is also available in a non-neural version that is based on Averaged Perceptron.

5.4 SCRIPTS Text Normalization

This is described as a normalization system that does text cleanup, transliteration and a bunch of other operations to control numbers, punctuation marks, repetitions and foreign text. https://github.com/rnd2110/SCRIPTS_Normalization.

5.5 SCRIPTS Summarizer

This component takes as input a query and a set of foreign language documents (and their translations) for any one of the MATERIAL languages that have been determined by CLIR to be relevant to the query. The summarizer produces a 100-word English word summary of the document for each component of the query. The github for the summarizer is here: https://github.com/kedz/scripts/. It must be called from within the SCRIPTS CLIR component.

5.6 SCRIPTS CLIR:

There are two components:

- A system to score documents with respect to a MATERIAL-format query for any one MATERIAL language
- A set of docker components that can be configured to score a MATERIAL-format query for a novel language

given a character normalization mapping table and a translation probability table for translation from that language to English

5.7 MARIAN NMT (https://marian-nmt.github.io/)

Described as a fast NMT system originally developed at EDIN. The software is free and open-source under the MIT license. Models have been trained for all the MATERIAL languages and can be trained using the toolkit. AFRL already uses the software.

5.8 ParaCrawl Code (https://github.com/paracrawl/)

This is an open-source code used to mine parallel corpora from the web (https://paracrawl.eu/). The pipeline uses a swappable tokenizer, sentence splitter, and baseline MT system so it can be extended to other languages. National Institute of Information and Communications Technology (NICT) ran our software to create a Japanese-English corpus: http://www.kecl.ntt.co.jp/icl/lirg/jparacrawl/. This project also received funding from the EU.

5.9 EDIN MT

We have docker images for NMT using Marian-NMT. Each docker image includes our trained models and all the necessary tools to create translations using the models. Necessary tools include data preprocessing, using Moses (http://statmt.org/moses/), Subword-NMT (https://github.com/rsennrich/subword-nmt), SentencePiece (https://github.com/google/sentencepiece), and python3.7(https://www. python.org/downloads/).

5.10 UMD MT

We have docker images for (a) Moses-based SMT systems, and (b) Sockeye-based NMT systems. Each of them can take as input text files in any of the project languages, preprocess them using Moses, Subword-NMT, and SentencePiece, and translate them into English (or vice versa).

5.11 ASR

We have separate ASR docker images for each of the project languages. Each of them can take WAV files as input and create output compressed triangle mesh (CTM) files. There is another script that converts the CTM files to a normal .txt file. Furthermore, the ASR docker images can output *n*-best lists for each utterance.

5.12 KWS

We have a language-independent KWS docker image that allows us to search for phrases in the recordings processed by the ASR docker images. The phrases must consist only of words in the ASR dictionary.

5.13 Audio Segmentation into Sentential Units

A component that has been trained using syntactic information to segment the output of ASR to produce sentence-like units. These are close in structure to what MT expects as input and thus, the component enables better translation of ASR output.

5.14 Morphological Analysis

We use the Adaptor Grammar framework (http://web.science.mq.edu.au/~mjohnson/code/naacl09.tgz) to train our segmentation models. We use GIZA++to (http://www.statmt.org/moses/giza/GIZA++.html) train alignment models. We use Stanza (https://github.com/stanfordnlp/stanza) to analyze the source languages for unsupervised cross-lingual training. We use the Percy Liang implementation of Brown Clusters (https://github.com/percyliang/brown-cluster) to train word-clustering models (optional). For the neural tagger, we use the Multilingual Roberta (XLM) (https://huggingface.co/transformers/model_doc/xlmroberta.html) to obtain contextual embeddings (optional).

5.15 Component for Determining Cross-Lingual Query-Relevance of Sentences

We can provide this component for all MATERIAL languages except Pashto. The component is trainable for new languages if parallel data, aligned at the sentence level, is available.

Performance of the component is improved with word alignments from phrase-based MT, but can operate with or without them.

6.0 CONCLUSIONS AND RECOMMENDATIONS

Here we present the reflections and recommendations of each team whose work was part of the SCRIPTS development process.

6.1 Speech Processing

The SCRIPTS ASR research findings highlighted that porting systems across domains is possible even for LRLs, as long as sufficient data for the language and approximate domain is available. Notably, however, we find that even untranscribed audio data can fill this data role, which helps transform the ASR possibilities for LRLs thanks to the ability to scrape audio from the web. At the same time, it is still important to address the impact of errors from the ASR system when using its outputs with downstream components.

A key area of future investigation for ASR on LRLs is to explore the degree of domain task porting that is possible when the availability of either text or audio data in the target language is limited. Our work indicates that the use of large volumes of untranscribed audio data is essential for ASR system development in the LRL context. Yet while unsupervised pre-training with such systems is a starting point for this approach, additional research is required to improve this type of ASR system development. Likewise, exploring how to link these large pre-trained systems with the particular needs of downstream systems will help yield more robust results.

6.2 MT

The MT work in this project reinforced the importance of robust data collection methods and pre-processing pipelines in order to quickly develop systems for new language pairs. It also demonstrated that neural models can provide useful translations even in LRL settings, as well as the benefits of tailoring MT systems to the distinct needs of upstream and downstream components. In the MATERIAL program, we confirmed this by improving results for other SCRIPTS component systems through techniques including dedicated data processing, providing alternative translation options, and incorporating query-specific constraints in translation.

A key area for future exploration is how to effectively measure the impact of MT errors on downstream components and users, which cannot always be reliably estimated at present.

Investigating how to support closer cooperation and tighter integration between MT and other components - even with minimal E2E training data available - would also enhance the efficacy and applicability of future MT work in this area.

6.3 Data Collection

Building software systems designed to locate answers to questions in LRLs requires collecting extremely large amounts of both speech and text data in each one. While this data can often be collected from the web, this approach first requires successfully building models to *identify* particular languages through training on true samples of the language desired. Likewise, systems that can successfully transcribe speech data from LRLs must also be developed. Finally, creating parallel language corpora for training MT models is also necessary. Building tools to collect and triage all of the described resources is important for any future work of this nature that focuses on LRLs.

6.4 Text Processing

The results of the SCRIPTS morphological analysis work illustrate that unsupervised approaches

- especially when guided by some form of linguistic priors - can obtain good performance for LRLs of diverse linguistic typology, from agglutinative to polysynthetic.

For morphological segmentation, the use of AGs allows the incorporation of linguistic priors in the form of either grammar definition or linguist-provided affixes. Moreover, for POS tagging, learning from multiple source languages via either decoding or projection had a positive impact. We also found that considering the stem as the unit of abstraction for alignments is powerful—particularly for morphologically complex languages.

A primary recommendation for future work in this area is to develop mechanisms for more robustly evaluating the impact of morphological analysis on downstream tasks like MT.

6.5 CLIR

From the perspective of CLIR, the timing of the MATERIAL program was exquisite. Prior to MATERIAL, a solid foundation of "traditional" approaches to CLIR - approaches built on sparse term representations - had already been developed. The concurrent turn toward neural DL across speech recognition, MT, and IR made the MATERIAL program an unprecedented opportunity to advance not just those individual technologies, but their integration and application to the CLIR task.

Our most important recommendation for CLIR would be to consider how to best capture and share the training data required by data hungry neural methods. While such training data need not be fully integrated across speech, translation, and retrieval, given the outsize impact that training data has on these systems' effectiveness, some consideration of those requirements will be both possible and necessary. Our work on the MATERIAL program leads us to suspect that there will be benefits to thinking of these data needs as closely coupled, just like the resulting systems. As a simple example, training IR systems can benefit from active learning; but the benefit for CLIR might be that much greater if the learning were to be performed on parallel rather than monolingual sources.

6.6 Summarization

Summarization research showed that an effective summarization model can be developed even when there is a total absence of annotated data for the query-focused task, thanks to synthetic data generation. Our work in the MATERIAL program demonstrated the benefit of synthetic data generation, especially when combined with rationale training for lexical queries; therefore, we extended this approach to conceptual queries. We were also able to develop abstractive cross-lingual summarization models by adapting our generation of synthetic data, through a process of adding "noise" to English input articles from an existing summarization dataset using back translation. We also showed how the generation of effective summaries in the LRL scenario depends on close integration with both ASR and MT.

Our primary recommendation going forward is to explore new task and evaluation designs that could encourage more creative solutions to summarization in order to maximize the utility of the summarization process for human end users. Based on our findings, it is clear that abstractive summarization could play a larger role in presenting more comprehensible summaries to a human end user, but the current evaluation framework did not reward solutions taking this approach.

7.0 REFERENCES

[1] Katharina Kann, Jesus Manuel Mager Hois, Ivan Vladimir Meza Ruiz, and Hinrich Schütze. Fortification of Neural Morphological Segmentation Models for Polysynthetic Minimal-Resource Languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 47–57, 2018.

[2] Rabih Zbib, Lingjun Zhao, Damianos Karakos, William Hartmann, Jay DeYoung, Zhongqiang Huang, Zhuolin Jiang, Noah Rivkin, Le Zhang, and Richard Schwartz. Neural-Network Lexical Translation for Cross-Lingual IR from Text and Speech. In *SI- GIR*, 2019.

[3] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The Kaldi Speech Recognition Toolkit. In *ASRU*, 2011.

[4] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur. Purely Sequence- Trained Neural Networks for ASR based on Lattice-Free MMI. In *Interspeech*, 2016.

[5] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-End Factor Analysis for Speaker Verification. *IEEE TASLP*, 19(4):788–798, 2010.

[6] George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny. Speaker Adaptation of Neural Network Acoustic Models using i-Vectors. In *ASRU*, 2013.

[7] Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur. Parallel Training of DNNs with Natural Gradient and Parameter Averaging. *arXiv preprint arXiv:1410.7455*, 2014.

[8] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *JMLR*, 15(1):1929–1958, 2014.

[9] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. *Interspeech 2019*, 2019.

[10] Andreas Stolcke. SRILM-An Extensible Language Modeling Toolkit. In *Seventh Inter-National Conference on Spoken Language Processing*, 2002.

[11] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Černockỳ, and Sanjeev Khudan-pur. Recurrent Neural Network Based Language Model. In *Interspeech*, 2010.

[12] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. Espnet: End-to-End Speech Processing Toolkit. Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, 2018-September:2207–2211, 2018.

[13] Anton Ragni, Qiujia Li, M. J. F. Gales, and Yongqiang Wang. Confidence Estimation and Deletion Prediction using Bidirectional Recurrent Neural Networks. In *2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, December 18-21, 2018*, pages 204–211. IEEE, 2018.

[14] Qiujia Li, Preben Ness, Anton Ragni, and Mark J. F. Gales. Bi-Directional Lattice Recurrent Neural Networks for Confidence Estimation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 6755–6759. IEEE, 2019.

[15] Alexandros Kastanos, Anton Ragni, and M. J. F. Gales. Confidence Estimation for Black Box Automatic Speech Recognition Systems using lattice Recurrent Neural Networks. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 6329–6333. IEEE, 2020.

[16] Anton Ragni and Mark J. F. Gales. Automatic Speech Recognition System Development in the "Wild". In B. Yegnanarayana, editor, *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*, pages 2217–2221. ISCA, 2018.

[17] Vimal Manohar, Hossein Hadian, Daniel Povey, and Sanjeev Khudanpur. Semi-Supervised Training of Acoustic Models Using Lattice-Free MMI. In *ICASSP*, 2018.

[18] Banriskhem Khonglah, Srikanth Madikeri, Subhadeep Dey, Hervé Bourlard, Petr Motlicek, and Jayadev Billa. Incremental Semi-Supervised Learning for Multi-Genre Speech Recognition. In *ICASSP*, 2020.

[19] Sree Hari Krishnan Parthasarathi and Nikko Strom. Lessons from Building Acoustic Models with a Million Hours of Speech. In *ICASSP*, 2019.

[20] Andrea Carmantini, Peter Bell, and Steve Renals. Untranscribed Web Audio for Low Resource Speech Recognition. In *Interspeech*, 2019.

[21] Karel Veselý, Mirko Hannemann, and Lukáš Burget. Semi-Supervised Training of Deep Neural Networks. In *ASRU*, 2013.

[22] Thomas Drugman, Janne Pylkkonen, and Reinhard Kneser. Active and Semi- Supervised Learning in ASR: Benefits on the Acoustic and Language Models. *arXiv preprint arXiv:1903.02852*, 2016.

[23] Electra Wallington, Benji Kershenbaum, Ondřej Klejch, and Peter Bell. On the Learning Dynamics of Semi-Supervised Training for ASR. In *Interspeech*, pages 716–720, 2021.

[24] Chau Luu, Peter Bell, and Steve Renals. Leveraging Speaker Attribute Information Using Multi-Task Learning for Speaker Verification and Diarization. In Interspeech, 2021.

[25] Kevin Burton, Akshay Java, Ian Soboroff, et al. The Icwsm 2009 Spinn3r Dataset. In Third Annual Conference on Weblogs and Social Media (ICWSM 2009). AAAI, 2009.

[26] Rico Sennrich and Martin Volk. MT-Based Sentence Alignment for OCR-Generated Parallel Texts. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas: Research Papers*, Denver, Colorado, USA, 31 October – 4 November 2010. Association for Machine Translation in the Americas.

[27] Víctor M. Sánchez-Cartagena, Marta Bañón, Sergio Ortiz-Rojas, and Gema Ramírez-Sánchez. Prompsit's Submission To WMT 2018 Parallel Corpus Filtering Shared Task. In *Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers*, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[28] Marco Lui and Timothy Baldwin. langid.py: An Off-The-Shelf Language Identification Tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[29] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Fed- erico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[30] Franz Josef Och. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July 2003. Association for Computational Linguistics.

[31] K. Bretonnel Cohen and Bob Carpenter, Editors. *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[32] Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable Modified Kneser-Ney Language Model Estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 690–696, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[33] David Chiang. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, Ann Arbor, Michigan, June 2005.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you Need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[35] Lei Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *CoRR*, abs/1607.06450, 2016.

[36] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818– 2826, 2016.

[37] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3th International Conference on Learning Representations*, 2015.

[38] Weijia Xu and Marine Carpuat. EDITOR: An Edit-Based Transformer with Repositioning for Neural Machine Translation with Soft Lexical Constraints. *Transactions of the Association for Computational Linguistics*, 9:311–328, 03 2021.

[39] Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. *Advances in Neural Information Processing Systems*, 32:11181–11191, 2019.

[40] Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. Marian: Fast Neural Machine Translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[41] Rico Sennrich and Biao Zhang. Revisiting Low-Resource Neural Machine Translation: A Case Study. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 211–221, Florence, Italy, July 2019. Association for Computational Linguistics.

[42] Tianchi Bi, Hao Xiong, Zhongjun He, Hua Wu, and Haifeng Wang. Multi-Agent Learning for Neural Machine Translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 856–865, Hong Kong, China, November 2019. Association for Computational Linguistics.

[43] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany, August 2016. Association for Computational Linguistics.

[44] Ofir Press and Lior Wolf. Using the Output Embedding to Improve Language Models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Computational*, pages 157–163. Association for Computational Linguistics, 2017.

[45] Weijia Xu, Xing Niu, and Marine Carpuat. Dual Reconstruction: A Unifying Objective for Semi-Supervised Neural Machine Translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2006–2020, Online, November 2020. Association for Computational Linguistics.

[46] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual Learning for Machine Translation. In *Advances in Neural Information Processing Systems 29*, pages 820–828. Curran Associates, Inc., 2016.

[47] Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 Conference on Machine Translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy, August 2019. Association for Computational Linguistics.

[48] Gideon Mendels, Erica Cooper, Victor Soto, Julia Hirschberg, Mark JF Gales, Kate M Knill, Anton Ragni, and Haipeng Wang. Improving Speech Recognition and Keyword Search for Low Resource Languages Using Web Data. In *Proceedings of INTERSPEECH*, pages 829–833, 2015.

[49] Tibor Kiss and Jan Strunk. Unsupervised Multilingual Sentence Boundary Detection. *Computational Linguistics*, 2006.

[50] Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. Adaptor Grammars: A Framework for Specifying Compositional Nonparametric Bayesian Models. In B. Schölkopf, J. Platt, and T. Hoffman, Editors, *Advances in Neural Information Processing Systems 19*, pages 641–648, Cambridge, MA, 2007. MIT Press.

[51] Jim Pitman. Exchangeable and Partially Exchangeable Random Partitions. *Probability Theory and Related Fields*, 102(2):145–158, 1995.

[52] Kairit Sirts and Sharon Goldwater. Minimally-Supervised Morphological Segmentation Using Adaptor Grammars. *Transactions of the Association for Computational Linguistics*, 1 (May):231–242, 2013.

[53] Jim Pitman, Marc Yor, et al. The Two-Parameter Poisson-Dirichlet Distribution Derived from a Stable Subordinator. *The Annals of Probability*, 25(2):855–900, 1997.

[54] Hemant Ishwaran and Lancelot F James. Generalized Weighted Chinese Restaurant Processes for Species Sampling Mixture Models. *Statistica Sinica*, pages 1211–1235, 2003.

[55] Mark Johnson. Unsupervised Word Segmentation for Sesotho Using Adaptor Grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[56] Howard I Aronson. *Georgian: A Reading Grammar, Corrected Edition*. Slavica Publishers, 1990.

[57] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A Python Natural Language Processing Toolkit for Many Human Languages. *arXiv preprint arXiv:2003.07082*, 2020.

[58] Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In *LREC*, volume 14, pages 1094–1101, 2014.

[59] Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational linguistics*, 29(1):19–51, 2003.

[60] Chris Dyer, Victor Chahuneau, and Noah A Smith. A Simple, Fast, and Effective Reparameterization of IBM model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, 2013.

[61] Martin F. Porter., Snowball: A Language for Stemming Algorithms. http://snowball.tartarus.org/texts/introduction.html, 2001.

[62] Edward Loper and Steven Bird. NLTK: The Natural Language Toolkit. *arXiv preprint cs/0205028*, 2002.

[63] Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. Token and Type Constraints for Cross-Lingual Part-of-Speech Tagging. *Transactions of the Association for Computational Linguistics*, 1:1–12, 2013.

[64] Michele Banko and Robert C Moore. Part-of-Speech Tagging in Context. In *Proceedings of the 20th international conference on Computational Linguistics*, page 556. Association for Computational Linguistics, 2004.

[65] Shen Li, Joao V Graça, and Ben Taskar. Wikily Supervised Part-of-Speech Tagging. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1389–1398. Association for Computational Linguistics, 2012.

[66] Jan Buys and Jan A. Botha. Cross-Lingual Morphological Tagging for Low-Resource Languages. In *In Proceedings of Human Language Technologies: The 2016 Annual Conference of the Association for Computational Linguistics, ACL'16*, pages 1954– 1964, Berlin, Germany, 2016.

[67] Long Duong, Paul Cook, Steven Bird, and Pavel Pecina. Simpler Unsupervised POS Tagging with Bilingual Projections. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 634–639, 2013.

[68] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[69] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF Models for Sequence Tagging. *arXiv preprint arXiv:1508.01991*, 2015.

[70] Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network. *arXiv preprint arXiv:1510.06168*, 2015.

[71] Barbara Plank, Anders Søgaard, and Yoav Goldberg. Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. In *In Proceedings of Human Language Technologies: The 2016 Annual Conference of the Association for Computational Linguistics, ACL'16*, pages 412–418, Berlin, Germany, August 2016. Association for Computational Linguistics.

[72] Xuezhe Ma and Eduard Hovy. End-to-End Sequence Labeling via Bi-Directional LSTM-CNNs-CRF. In *In Proceedings of Human Language Technologies: The 2016 Annual Conference of the Association for Computational Linguistics, ACL'16*, pages 1064– 1074, Berlin, Germany, August 2016. Association for Computational Linguistics.

[73] Ryan Cotterell and Georg Heigold. Cross-Lingual Character-Level Neural Morphological Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 748–759, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[74] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June 2016. Association for Computational Linguistics.

[75] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceed- ings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, 2020.

[76] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479, 1992.

[77] Percy Liang. *Semi-Supervised Learning for Natural Language*. PhD thesis, Massachusetts Institute of Technology, 2005.

[78] Željko Agić, Dirk Hovy, and Anders Søgaard. If All You Have is a Bit of the Bible: Learning POS Taggers for Truly Low-Resource Languages. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 268–272, 2015.

[79] Željko Agić, Anders Johannsen, Barbara Plank, Héctor Martínez Alonso, Natalie Schluter, and Anders Søgaard. Multilingual Projection for Parsing Truly Low-Resource Languages. *Transactions of the Association for Computational Linguistics*, 4:301–312, 2016.

[80] Barbara Plank and Željko Agić. Distant Supervision from Disparate Sources for Low-Resource Part-of-Speech Tagging. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 614–620, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[81] Thomas Mayer and Michael Cysouw. Creating a Massively Parallel Bible Corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3158–3163, 2014.

[82] Kareem Darwish and Douglas W Oard. Probabilistic Structured Query Methods. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 338–344. ACM, 2003.

[83] Edward A Fox and Joseph A Shaw. Combination of Multiple Searches. *NIST Special Publication*, 243, 1994.

[84] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*, 2016.

[85] Peter H Schönemann. A Generalized Solution of the Orthogonal Procrustes Problem.*Psychometrika*, 31(1):1–10, 1966.

[86] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre- training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, June 2019.

[87] Zi-Yi Dou and Graham Neubig. Word Alignment by Fine-Tuning Embeddings on Parallel Corpora. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, April 2021.

[88] Evan Sandhaus. The New York Times Annotated Corpus LDC2008T19, 2008.

[89] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*, 2013.

[90] Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. Document Expansion by Query Prediction. *arXiv preprint arXiv:1904.08375*, 2019.

[91] Zhuyun Dai and Jamie Callan. Context-Aware Sentence/Passage Term Importance Estimation for First Stage Retrieval. *arXiv preprint arXiv:1910.10687*, 2019.

[92] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. Indri: A Language Model-Based Search Engine for Complex Queries. In *Proceedings of the International Conference on Intelligence Analysis*, volume 2, pages 2–6. Citeseer, 2005.

[93] Peilin Yang, Hui Fang, and Jimmy Lin. Anserini: Enabling the Use of Lucene for Information Retrieval Research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256, 2017.

[94] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at TREC-3. In *Proceedings of the Third Text Retrieval Conference*. National Institute OF Standards & Technology, 1995.

[95] David R. H. Miller, Tim Leek, and Richard M. Schwartz. A Hidden Markov Model Information Retrieval System. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, page 214–221, New York, NY, USA, 1999. Association for Computing Machinery.

[96] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *arXiv preprint arXiv:1312.3005*, 2013.

[97] Jay M Ponte and W Bruce Croft. A Language Modeling Approach to Information Retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281, 1998.

[98] Chengxiang Zhai and John Lafferty. A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):179–214, 2004.

[99] Donald Metzler and W Bruce Croft. A Markov Random Field Model for Term Dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 472–479, 2005.

[100] Doğan Can and Murat Saraclar. Lattice Indexing for Spoken Term Detection. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(8):2338–2347, 2011.

[101] Murat Saraclar and Richard Sproat. Lattice-Based Search for Spoken Utterance Retrieval. In *NAACL*, 2004.

[102] Omar Khattab and Matei Zaharia. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48, 2020.

[103] Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. PACRR: A Position- Aware Neural IR Model for Relevance Matching. *arXiv preprint arXiv:1704.03940*, 2017.

[104] Ryan McDonald, Georgios-Ioannis Brokos, and Ion Androutsopoulos. Deep Relevance Ranking Using Enhanced Document-Query Interactions. *arXiv preprint arXiv:1809.01682*, 2018.

[105] Rui Zhang, Caitlin Westerfield, Sungrok Shim, Garrett Bingham, Alexander Fabbri, William Hu, Neha Verma, and Dragomir Radev. Improving Low-Resource Cross-Lingual Document Retrieval by Reranking with deep Bilingual Representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3173–3179, July 2019.

[106] Saurabh Kulshreshtha, Jose Luis Redondo Garcia, and Ching-Yun Chang. Cross-lingual Alignment Methods for Multilingual BERT: A Comparative Study. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 933–942, November 2020.

[107] Steven Cao, Nikita Kitaev, and Dan Klein. Multilingual Alignment of Contextual Word Representations. In *International Conference on Learning Representations*, 2020.

[108] Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. CEDR: Contextualized Embeddings for Document Ranking. *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019.

[109] S. Wu. *Data Fusion in Information Retrieval*. Springer, 2012.

[110] JC de Borda. Mémoire sur les élections au scrutin. *Histoire de l'Academie Royale des Sciences pour 1781 (Paris, 1784)*, 1784.

[111] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, page 758–759, New York, NY, USA, 2009. Association for Computing Machinery.

[112] Joao Palotti, Harrisen Scells, and Guido Zuccon. Trectools: An Open-Source Python Library for Information Retrieval Practitioners Involved in Trec-Like Campaigns. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1325–1328, 2019.

[113] Stephen E Robertson. The Probability Ranking Principle in IR. *Journal of Documen tation*, 1977.

[114] Richard Schwartz, John Makhoul, Lee Tarlin, and Damianos Karakos. What Set of Documents to present to an analyst? In *Proceedings of the Workshop on Cross- Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 52–57, Marseille, France, May 2020. European Language Resources Association.

[115] Peter Emerson. The Original Borda Count and Partial Voting. *Social Choice and Welfare*, 40(2):353–358, 2013.

[116] Eunah Cho, Jan Niehues, and Alex Waibel. NMT-based Segmentation and Punctuation Insertion for Real-Time Spoken Language Translation. In *Proceedings of Interspeech 2017*, pages 2645–2649, 2017.

[117] Evgeny Matusov, Dustin Hillard, Mathew Magimai-doss, Dilek Hakkani-tur, Mari Ostendorf, and Hermann Ney. Improving Speech Translation with Automatic Boundary Prediction. In *Proceedings of Interspeech*, pages 2449–2452, 01 2007.

[118] Sharath Rao, Ian Lane, and Tanja Schultz. Optimizing Sentence Segmentation for Spoken Language Translation. In *Proceedings of Interspeech*, 2007.

[119] Surabhi Gupta, Ani Nenkova, and Dan Jurafsky. Measuring Importance and Query Rel evance in Topic-Focused Multi-Document Summarization. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 193–196, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[120] Tal Baumel, Matan Eyal, and Michael Elhadad. Query Focused Abstractive Summarization: Incorporating Query Relevance, Multi-Document Coverage, and Summary Length Constraints into Seq2Seq Models. *CoRR*, abs/1801.07704, 2018.

[121] Jörg Tiedemann. Parallel Data, Tools and Interfaces in OPUS. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, Editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 2214–2218. European Language Resources Association (ELRA), 2012.

[122] David Kamholz, Jonathan Pool, and Susan Colowick. PanLex: Building a Resource for Panlingual Lexical Translation. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, Editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[123] Sarthak Jain and Byron C. Wallace. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[124] Xing Niu, Michael Denkowski, and Marine Carpuat. Bi-directional Neural Machine Translation with Synthetic Parallel Data. *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, 2018.

[125] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, 2007.

[126] Kenneth Heafield. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, July 2011. Association for Computational Linguistics.

[127] Thang Luong, Hieu Pham, and Christopher D. Manning. Bilingual Word Representations with Monolingual Quality in Mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, Denver, Colorado, June 2015. Association for Computational Linguistics.

[128] Omer Levy, Anders Søgaard, and Yoav Goldberg. A Strong Baseline for Learning Cross-Lingual Word Embeddings from Sentence Alignments. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017.

[129] Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word Translation without Parallel Data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, 30 April - 3 May 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[130] Jinxi Xu and Ralph Weischedel. Cross-Lingual Information Retrieval Using Hidden Markov Models. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora: Held in Conjunction with the 38th Annual Meeting of the Association for Computational Linguistics- Volume 13*, EMNLP '00, page 95–103, USA, 2000. Association for Computational Linguistics.

[131] Georgiana Dinu and Marco Baroni. Improving Zero-Shot Learning by Mitigating the

Hubness Problem. In the *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.

[132] Milos Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. On the Existence of Obstinate Results in Vector Space Models. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '10, page 186–193, New York, NY, USA, 2010. Association for Computing Machinery.

[133] Franz Josef Och and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

[134] Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. Better Word Alignments with Supervised ITG Models. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 923–931, Suntec, Singapore, August 2009. Association for Computational Linguistics.

[135] Yanda Chen, Chris Kedzie, Suraj Nair, Petra Galuscakova, Rui Zhang, Douglas Oard, and Kathleen McKeown. Cross-Language Sentence Selection via Data Augmentation and Rationale Training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3881–3895, Online, August 2021. Association for Computational Linguistics.

[136] Michel Simard, Nicola Ueffing, Pierre Isabelle, and Roland Kuhn. Rule-Based Translation with Statistical Phrase-Based Post-Editing. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 203–206, 2007.

[137] Rajen Chatterjee, Matteo Negri, Raphael Rubino, and Marco Turchi. Findings of the WMT 2018 Shared Task on Automatic Post-Editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 710–725, Belgium, Brussels, October 2018. Association for Computational Linguistics.

[138] David Wan, Chris Kedzie, Faisal Ladhak, Marine Carpuat, and Kathleen McKeOwn. Incorporating Terminology Constraints in Automatic Post-Editing. *arXiv preprint arXiv:2010.09608*, 2020.

[139] Jessica Ouyang, Boya Song, and Kathy McKeown. A Robust Abstractive System for Cross-Lingual Summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2025–2031, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[140] Xiaojun Wan, Huiying Li, and Jianguo Xiao. Cross-Language Document Summarization Based on Machine Translation Quality Prediction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 917–926, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[141] Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[142] Evan Sandhaus. The New York Times Annotated Corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752, 2008.

[143] Yanda Chen, Chris Kedzie, Suraj Nair, Petra Galuscáková, Rui Zhang, Douglas W. Oard, and Kathleen R. McKeown. Cross-Language Sentence Selection via Data Augmentation and Rationale Training. *CoRR*, abs/2106.02293, 2021.

[144] Matt Post. A Call for Clarity in Reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[145] Matt Post and David Vilar. Fast Lexically Constrained Decoding with Dynamic Beam Allocation for Neural Machine Translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[146] Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[147] Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. Automatic Evaluation of Translation Quality for Distant Language Pairs. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 944–952, Cambridge, MA, October 2010. Association for Computational Linguistics.

[148] C. Biemann, G. Heyer, U. Quasthoff, and M. Richter. The Leipzig Corpora Collection: Monolingual Corpora of Standard Size. In *Proceedings of Corpus Linguistics 2007*, 2007.

[149] Michael McCandless. Accuracy and Performance of Google's Compact Language Detector. http://blog.mikemccandless.com/2011/10/accuracy-and-performance-of-googles.html, 2010.

[150] K. Hornik, P. Mair, W. Geiger, J. Rauch, C. Buchta, and I. Feinerer. The Textcat Package for N-Gram Based Text Categorization in R. *Journal of Statistical Software*, 52(6):1–17, 2013.

[151] Breck Baldwin and Bob Carpenter. Lingpipe. *Available from World Wide Web:* http://alias-i. com/lingpipe, 2003.

[152] Hoifung Poon, Colin Cherry, and Kristina Toutanova. Unsupervised Morphological Segmentation with Log-Linear Models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL'09*, pages 209–217, Boulder, Colorado, June 2009. Association for Computational Linguistics.

[153] Karthik Narasimhan, Regina Barzilay, and Tommi Jaakkola. An Unsupervised Method for Uncovering Morphological Chains. In *Twelfth AAAI Conference on Artificial Intelligence*, 2015.

[154] Christos Christodouloupoulos and Mark Steedman. A Massively Parallel Corpus: The Bible in 100 Languages. *Language Resources and Evaluation Conference (LREC)*, 49(2):375–395, 2015.

[155] Daniel Zeman, Martin Popel, Milan Straka, and et al. CoNLL 2017 shared task: Multi-Lingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[156] Ramy Eskander, Cass Lowry, Sujay Khandagale, Francesca Callejas, Judith Klavans, Maria Polinsky, and Smaranda Muresan. Minimally-Supervised Morphological Segmentation using adaptor Grammars with Linguistic Priors. In *ACL Findings*, 2021.

[157] Aliaksei Severyn and Alessandro Moschitti. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382. ACM, 2015.

[158] Sunil Mohan, Nicolas Fiorini, Sun Kim, and Zhiyong Lu. Deep Learning for Biomedical Information Retrieval: Learning Textual Relevance from Click Logs. *BioNLP 2017*, pages 222–231, 2017.

[159] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, 2015.

[160] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word Translation without Parallel Data. In *ICLR*, 2018.

[161] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align And Translate. In *ICLR*, 2015.

[162] Antonio Valerio Miceli Barone, Jindřich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. Deep Architectures for Neural Machine Translation. In *Proceedings of the Second Conference on Machine Translation*, Copenhagen, Denmark, 2017. Association for Computational Linguistics.

[163] J Scott McCarley. Should we Translate the Documents or the Queries in Cross-Language Information Retrieval? In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 208–214. Association for Computational Linguistics, 1999.

[164] Martin Franz, J Scott McCarley, and Salim Roukos. Ad hoc and Multilingual Information Retrieval at IBM. *NIST Special Publication SP*, pages 157–168, 1999.

[165] Ilya Zavorin, Aric Bills, Cassian Corey, Michelle Morrison, Audrey Tong, and Richard Tong. Corpora for Cross-Language Information Retrieval in Six Less-Resourced Languages. In *Proceedings of the Workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 7–13, May 2020.

[166] Carol Peters and Martin Braschler. European Research Letter: Cross-Language System Evaluation: The CLEF Campaigns. *Journal of the American Society for Information*

Science and Technology, 52(12):1067–1072, 2001.

[167] Victor Lavrenko, Martin Choquette, and W. Bruce Croft. Cross-Lingual Relevance Models. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '02, page 175–182, 2002.

[168] Ivan Vulić and Marie-Francine Moens. Monolingual and Cross-Lingual Information Retrieval Models Based on (Bilingual) Word Embeddings. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 363–372, 2015.

[169] Kalervo Järvelin and Jaana Kekäläinen. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[170] Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT Rediscovers the Classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, July 2019.

[171] Wietse de Vries, Andreas van Cranenburgh, and Malvina Nissim. What's so Special about BERT's Layers? A Closer Look at the NLP Pipeline in Monolingual and Multilingual Models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4339–4350, November 2020.

[172] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does BERT Learn about the Structure of Language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, July 2019.

[173] Shijie Wu and Mark Dredze. Are all Languages Created Equal in Multilingual BERT? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120– 130, July 2020.

[174] Caitlin Christianson, Jason Duncan, and Boyan A. Onyshkevych. Overview of the DARPA LORELEI Program. *Machine Translation*, 32(1-2):3–9, 2018.

[175] J. Kittler, M. Hatef, Robert Duin, and Jiri Matas. On Combining Classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20:226–239, 01 2002.

[176] Jonathan G Fiscus. A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 347–354. IEEE, 1997.

[177] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A Human Generated Machine Reading Comprehension Dataset. *CoRR*, abs/1611.09268, 2016.

[178] Marcello Federico and Nicola Bertoldi. Statistical Cross-Language Information Retrieval using n-best Query Translations. In Kalervo Järvelin, Micheline Beaulieu, Ricardo A. Baeza-Yates, and Sung-Hyon Myaeng, editors, *SIGIR 2002: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 11-15, 2002, Tampere, Finland*, pages 167–174. ACM, 2002.

[179] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[180] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in Pre-Training Distributed Word Representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).

[181] David Wan, Zhengping Jiang, Chris Kedzie, Elsbeth Turcan, Peter Bell, and Kathy McKeown. Subtitles to Segmentation: Improving Low-Resource Speech-to-Text Translation Pipelines. In *Proceedings of the Workshop on Cross-Language Search and Summarization of Text and Speech (CLSSTS2020)*, pages 68–73, Marseille, France, May 2020. European Language Resources Association.

[182] Pierre Lison and Jörg Tiedemann. OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA).

[183] Lev Pevzner and Marti A. Hearst. A Critique and Improvement of an Evaluation Metric for Text Segmentation. *Computational Linguistics*, 28(1):19–36, 2002.

[184] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. A Text Feature Based Automatic Keyword Extraction Method for Single Documents. In *European Conference on Information Retrieval*, pages 684–691. Springer, 2018.

[185] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*, abs/1907.11692, 2019.

[186] David Kamholz, Jonathan Pool, and Susan M. Colowick. Panlex: Building a Resource Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pages 3145–3150. European Language Resources Association (ELRA), 2014.

[187] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[188] Marcin Junczys-Dowmunt and Roman Grundkiewicz. MS-UEdin submission to the WMT2018 APE shared task: Dual-Source Transformer for Automatic Post-Editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 822–826, Belgium, Brussels, October 2018. Association for Computational Linguistics.

[189] Jiatao Gu, Changhan Wang, and Jake Zhao. Levenshtein Transformer. *CoRR*, abs/1905.11006, 2019.

[190] David Wan, Chris Kedzie, Faisal Ladhak, Marine Carpuat, and Kathleen R. McKeown. Incorporating Terminology Constraints in Automatic Post-Editing. *CoRR*, abs/2010.09608, 2020.

[191] Paul Over and James Yen. An Introduction to DUC 2004 Intrinsic Evaluation of Generic New Text Summarization Systems. In *Proceedings of DUC 2004 Document*

Understanding Workshop, Boston, Massachusetts, 2004.

[192] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[193] Nils Reimers and Iryna Gurevych. Making Monolingual Sentence Embeddings Multilingual Using Knowledge Distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online, November 2020. Association for Computational Linguistics.

[194] David Wan. Methods for Cross-Language Search and Summarization for Low-Resource Languages. Master's Thesis, Columbia University, 7 2021. Advisor: Kathleen McKeown.

[195] Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In Text Summarization Branches Out, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[196] Nils Reimers and Iryna Gurevych. Making Monolingual Sentence Embeddings Multilingual Using Knowledge Distillation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 4512–4525, Online, November 2020. Association for Computational Linguistics.

[197] David Wan. Methods for Cross-Language Search and Summarization for Low- Resource Languages. Master's thesis, Columbia University, 7 2021. Advisor: Kathleen McKeown.

8.0 LIST OF ABBREVIATIONS, ACRONYMS AND SYMBOLS

AFRL	Air Force Research Laboratory
AG	Adaptor Grammar
AGIC	Application Gateway Ingress Controller
AG-LI	Adaptor Grammar Language Independent
AG-SS	Adaptor Grammars Scholar-Seeded
AM	Acoustic Model
AMT	Amazon Mechanical Turk
ANN	Approximate Nearest Neighbor
ANTLR	ANother Tool for Language Recognition
APE	Automatic Post-Editing
API	Application Programming Interface
AQWV	Average Query Weighted Value
ASR	Automatic Speech Recognition
AT	Autoregressive Transformer
AUC	Area Under the Curve
AWS	Amazon Web Service
AWESOME	Aligning Word Embedding Spaces of Multilingual Encoders
BERT	Bidirectional Encoder Representations from Transformers
BestDA	Best Data-Augmentation
BestMTT	Best Multi-Task Training
BNB	Babel Narrow Band
BNF	Backus Naur Form Notation
BP	Base Period
BiLSTM	Bidirectional Long Short-Term Memory
BLEU	Bilingual Evaluation Understudy Score
BPE	Byte-Pair Encoding
BPR	Boundary Precision and Recall
BT	Blog Text
CFG	Context Free Grammar
CLD	Compact Language Detector
CEDR	Contextualized Embeddings for Document Ranking
CLEF	Cross-Language Evaluation Forum

CLIR	Cross-Language Information Retrieval
CL-SR	Cross Language Speech Retrieval
CNN-TDNNF	Convolutional Neural Network/ Time-Delay Neural Network Factorization
CoNLL	Computational Natural Language Learning
CPR	Constraint Preservation Rate
CRF	Conditional-Random Fields
CS	Conversational Speech
CTM	Compressed Triangle Mesh
CTS	Conversational Telephone Speech
CUED	Cambridge
DBQT	Dictionary-Based Query Translation
DET	Detection Error Tradeoff
DL	Deep Learning
DRMM	Deep Relevance Matching Model
DsDs	Distant Supervision from Disparate Sources
DUC	Document Understanding Conference
E2E	End-To-End
EDIN	Edinburgh
EDINMT	Edinburgh Machine Translation
EDITOR	Edit-Based Transformer with Repositing
ESPnet	Efficient Spatial Pyramid Network
EU	European Union
FA-Accept	False Alarm to Accept
FIRE	Forum for Information Retrieval Evaluation
GloVe	Global Vectors for Word Representation
GPU	Graphic Processing Unit
GT	Ground Truth
GT Irrel	Ground Truth Irrelevant
GT Rel	Ground-Truth Relevant
HDF5	Hierarchical Data Format 5
HMM-DNN	Hybrid Hidden Markov/Deep Neural Network
IARPA	Intelligence Advanced Research Projects Activity

IDF	Inverse Document Frequency
IR	Information Retrieval
ISI	Information Sciences Institute
iSST	Incremental Semi-Supervised Training
IV	In-Vocabulary
IWSLT	International Workshop on Spoken Language Translation
JAR	Java Archive
JSON	JavaScript Object Notation
KWS	Keyword Spotting
LF-MMI	Lattice-Free Maximum Mutual Information
LDA	Latent Dirichlet Allocation
LDC	Linguistic Data Consortium
LevT	Levenshtein Transformer
LI	Language Independent
LRL	Low Resource Language
LM	Language Model
LS	Language Specific
LSTM	Long Short-Term Memory
MAP	Mean Average Precision
MATERIAL	Machine Translation for English Retrieval of Information in Any Language
mBERT	Multilingual Bidirectional Encoder Representations
MERT	Minimum Error Rate Training
MFCC	MelFrequency Cepstral Coefficient
MLM	Masked Language Modeling
MLP	Multilayer Perceptron
MNB	MATERIAL Narrow Band
MQWV	Maximum Query Weighted Value
MS MARCO	Microsoft Machine Reading Comprehension
MST	Multi-Source Transformer Instantiation
MT	Machine Translation
MUSE	Multilingual Unsupervised and Supervised Embeddings
MWB	MATERIAL Wideband

NAACL	North American Chapter of the Association for Computational Linguistics
NAR	Non-Autoregressive
NAT	Non-Autoregressive Transformer
NB	Narrow Band
NB	News Broadcast
NCE	Normalized Cross Entropy
NICT	National Institute of Information and Communications Technology
NLTK	Natural Language Toolkit
NMT	Neural Machine Translation
NNLTM	Neural-Network Lexical Translation Model
NSP	Next Sentence Prediction
NT	Text Types
NYT	New York Times
OOL	Out-of-Language
OOV	Out-of-Vocabulary
OP1	Option Period 1
OPUS	Open Parallel Corpus
PACRR	Position-Aware Convolutional Recurrent Relevance Matching
PBMT	Phrase-Based Machine Translation
PCFG	Probabilistic Context-Free Grammar
POS	Part-of-Speech
POSIT	POoled SImilariTy
PSI	Parallel Sentence Identification
PSQ	Probabilistic Structured Queries
PTB	Penn Treebank
PTO	Probability of Term Occurrence
QFS	Query-focused summarization
QLM	Query Likelihood Model
QST	Query-Specific Thresholds
RIBES	Rank-based Intuitive Bilingual Evaluation Score
RNN-LM	Recurrent Neural Network-Based Language Model
RRF	Reciprocal Rank Fusion
ROC	Receiver Operating Characteristic

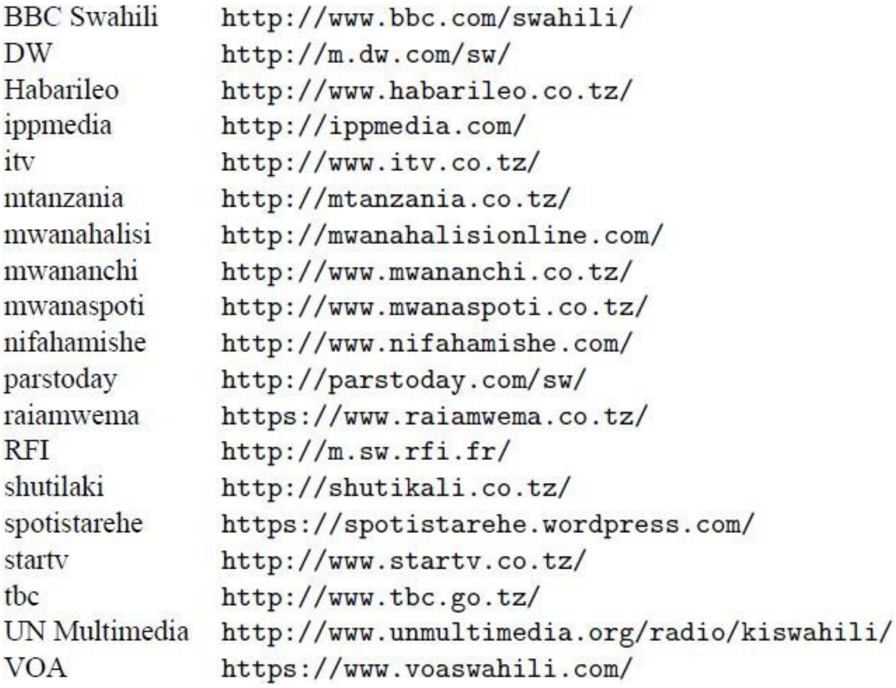
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
RRF	Reciprocal Rank Fusion
RT	Rationale Training
S2S	Sequence to Sequence
SCRIPTS	System for CRoss-language Information Processing, Translation and Summarization
SDM	Sequential Dependence Model
SECLR	Supervised Embedding-based Cross-Lingual Relevance
SECLR-RT	Supervised Embedding-based Cross-Lingual Relevance with Rationale Training
SID-SGNS	Sentence Identification-Skip-Grams with Negative Sampling
SLT	Spoken Language Translation
SMT	Statistical Machine Translation
SOTA	State-of-The-Art
SRILM	SRI Language Modeling
SST	Semi-Supervised Training
STO	Sum-To-One
STTT	Speech-to-Text Translation
SVM	Support Vector Machine
TB	Topical Broadcast
TD-to-Miss	True Detections to Miss
TDNN	Time Delay Neural Network
TED	Technology, Entertainment, Design
LIUM	Laboratoire d'Informatique de l'Université du Maine
TER	Translation Error Rate
TF-IDF	Term Frequency-Inverse Document Frequency
TREC	Text Retrieval Conference
TLM	Translation Language Modeling
UD	Universal Dependencies
UMD	University of Maryland
URL	Uniform Resource Locators
VAD	Voice Activity Detection
VBERT	Vanilla mBERT

Vtt	Video Text Tracks
WB	Wide-Band
WD	Windowdiff
WER	Word-Error Rate
WFST	Weighted Finite State Transducer
WMT	Workshop on Machine Translation
XLM	Cross-Lingual LM
XLM-R	XLM RoBERTa
XML	Extensible Markup Language

APPENDIX A - List of News Sites

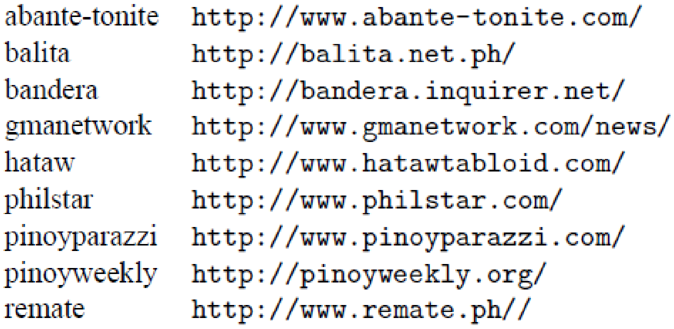
List of Swahili and Tagalog News Websites

Table A1. Swahili News Sites



BBC Swahili	http://www.bbc.com/swahili/
DW	http://m.dw.com/sw/
Habarileo	http://www.habarileo.co.tz/
ippmedia	http://ippmedia.com/
itv	http://www.itv.co.tz/
mtanzania	http://mtanzania.co.tz/
mwanahalisi	http://mwanahalisionline.com/
mwananchi	http://www.mwananchi.co.tz/
mwanaspoti	http://www.mwanaspoti.co.tz/
nifahamishe	http://www.nifahamishe.com/
parstoday	http://parstoday.com/sw/
raiamwema	https://www.raiamwema.co.tz/
RFI	http://m.sw.rfi.fr/
shutilaki	http://shutikali.co.tz/
spotistarehe	https://spotistarehe.wordpress.com/
startv	http://www.startv.co.tz/
tbc	http://www.tbc.go.tz/
UN Multimedia	http://www.unmultimedia.org/radio/kiswahili/
VOA	https://www.voaswahili.com/

Table A2. Tagalog News Sites



abante-tonite	http://www.abante-tonite.com/
balita	http://balita.net.ph/
bandera	http://bandera.inquirer.net/
gmanetwork	http://www.gmanetwork.com/news/
hataw	http://www.hatawtabloid.com/
philstar	http://www.philstar.com/
pinoyparazzi	http://www.pinoyparazzi.com/
pinoyweekly	http://pinoyweekly.org/
remate	http://www.remate.ph//

APPENDIX B - SCRIPTS Normalization Parameters

SCRIPTS Normalization Parameters

The SCRIPTS Normalization system[67] accepts a set of parameters that control the normalization process. The parameters are as follows:

- language:string (case-insensitive): Material codes (e.g., 1A), ISO codes (e.g., SWA) and full language names (e.g., Swahili) are all accepted inputs.
- text:string
- letters_to_keep:string (case-sensitive): Letters needed to be kept, overwrites the re- moval of vowels, diacritics, non-alphabet characters and built-in language mappings –"" means do not use this feature.
- letters_to_remove:string (case-sensitive): Letters needed to be removed – "" means donot use this feature.
- lowercase:boolean
- remove_repetitions_count:int: The maximum number of allowed character repetitions(in a sequence), e.g., when set to 2, "mannner" changes to "manner" – 0 means do notuse this feature (after the built-in mapping, lower-casing and removal of extras (e.g., non-zero width joiners) and before any other operations).
- remove_punct:boolean: Covers both punctuation marks and symbols
- remove_digits:boolean
- remove_vowels:boolean: Does not cover the short-vowel diacritics in Pashto and Farsi, and does not affect non-alphabet characters of the underling languages
- remove_diacritics:boolean
- remove_spaces:boolean
- remove_apostrophe:boolean
- copy_through:boolean: When set to True, none of the foreign letters gets omitted.
- keep_romanized_text:boolean: This argument works when the language has a non- Latin script (Bulgarian and Pashto). When set to True, none of the letters of the Ro- manized Bulgarian script), in the case of Bulgarian, and the Romanized Pashto script,in the case of Pashto) gets omitted. When set to False, the Romanized Bulgarian letters are transliterated into the Cyrillic script, in the case of Bulgarian, and the romanized Pashto letters are omitted (Notice that transliteration in Pashto is not supported).

[67] https://github.com/rnd2110/SCRIPTS_Normalization

APPENDIX C – SCRIPTS NCTE Final Tech Report (Added Information)

SCRIPTS NCTE FINAL REPORT

December 2021

Note – The Table of Contents for this appendix follow the numbering for this whole tech report.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

This report describes our technical approaches and results for System for CRoss-language Information Processing, Translation and Summarization (SCRIPTS) for the No-Cost Technical Extension (NCTE) period of the Intelligence Advanced Research Projects Activity (IARPA) Machine Translation (MT) for English Retrieval of Information in Any Language Multilingual Bi-directional Encoder Representations from Transformers (BERT) (MATERIAL) program.

SCRIPTS consists of components for Automatic Speech Recognition (ASR) and MT in order to pre-process the text and speech corpora provided as part of the program. It also includes a text processing component that performs morphological analysis. In user-facing mode, given a query, SCRIPTS' Cross-Language Information Retrieval (CLIR) returns relevant documents, while Summarization generates textual summaries of each document to help an analyst confirm which documents returned by CLIR are actually relevant. Over the course of program, we implemented models for nine different languages: Somali, Swahili and Tagalog in the Base Period (BP), Bulgarian, Lithuanian and Pashto in Option Period (OP) 1, and Farsi, Kazakh and Georgian in OP2.

To address the low-resource scenario, our novel approach in SCRIPTS featured implicit and explicit integration within and across components. We use a fail-soft approach, where each component implemented multiple, complementary approaches to the task; these components were then integrated using system combination. For ASR, we developed two approaches, one at Cambridge University (CUED) and one at the University of Edinburgh (EDIN); here, system combination yielded improvements in scores over individual systems. For MT, we developed two neural approaches, one at the University of Edinburgh and another at the University of Maryland (UMD). UMD also implemented a statistical machine translation (SMT) approach. During evaluations, we ran all three MT systems on the evaluation corpora, and all three sets of results were saved and used by both CLIR and Summarization. CLIR used a large variety of technical approaches and system combination to merge the resulting rankings. While CLIR focused on finding relevant documents given the query, the task for Summarization was to find relevant sentences within the documents returned by CLIR. Like CLIR, Summarization also used multiple approaches for the problem, exploring both unsupervised and supervised methods.

Other key features of our approach included tight interaction between different components. For example, CLIR relies on an interaction with ASR and MT in order to handle search over speech in low-resource languages. Summarization also relies on results from CLIR as well as interaction with MT in order to ensure that at least one of its summary sentences contains the query word(s).

The report is structured as follows. First, we provide a summary of major differences between components across the BP, OP1 OP2 and the NCTE. Then we provide detailed information on the technical approaches taken by each component during the NCTE followed by a section providing results, again segmented by system component.

2.0 INTRODUCTION

The SCRIPTS development teams worked both in tandem and in parallel to develop, test, share and refine their respective components of the overall system across all phases of the work. Below, we highlight the key approaches and results for each team during the NCTE.

2.1 ASR

The principal function of the SCRIPTS ASR work was to process audio documents in order to make them sufficiently usable for MT and CLIR. In both the BP and OP1, the acoustic model (AM) parameters were augmented with multi-lingual bottleneck features. A main distinction between the BP and OP1 for this work was the use of large quantities of un-transcribed audio that had been scraped from the web. During OP2, more advanced approaches for using this data, in the form of the lattice- based semi-supervised training were refined and deployed. In the NCTE, we focused on new methods for estimating confidence and for "deciphering" target language speech from the output of a multi-lingual phone recognizer.

2.2 MT

Each period increased adaptation of MT to program goals and integration with surrounding systems. In the BP, we developed MT systems specifically adapted to the low-resource language (LRL) context of MATERIAL via back-translation and other features. In OP1, we experimented with forms of *n*-best output to improve indexing and adapt to speech input. Because the languages in OP1 were also higher resource, we were able to improve the system by exploiting changes in neural MT architectures. In OP2, we found that adding query-driven translation improved acceptance of summaries. At the same time, we found that Kazakh presented an interesting multilingual challenge, given that Russian-Kazakh data is more readily available than Kazakh-English data. In the NCTE, we focused creating a Farsi→English translation system using the recipe developed for English→German translation but found our model may have been overfitted to German. We also experimented with the impact of retranslating documents with query words as constraints to produce summaries.

2.3 Text Processing

During the BP, we developed MorphAGram, a framework for the unsupervised morphological segmentation of a diverse set of languages, including the automatic tailoring of grammars for unseen languages. During OP1 and OP2, we enhanced MorphAGram to allow for the incorporation of linguistic priors, in the form of either grammar definition or linguist-provided affixes. During OP1, we also developed unsupervised part-of-speech (POS) taggers using cross-lingual projection using both an averaged perceptron model and a neural model. In OP2, this POS tagger was enhanced through the addition of two key features. First, we developed support for learning from multiple source languages. Second, we made it possible for the system to use both word stems and morphemes as the unit of abstraction during the alignment process. In the NCTE, we focused on improvements to unsupervised POS tagging.

2.4 CLIR

In the BP, we developed the core capabilities for CLIR, including query analysis, ranking using both MT for 1-best translation and Probabilistic Structured Queries (PSQ) for *n*-best translation, rank-based late fusion for combining system results, cutoff tuning, and formative evaluation. In OP1, we extended our *n*-best approaches both to *n*-best ASR using Keyword Spotting (KWS) and through *n*-best Statistical MT and *n*-best Neural MT (NMT) and a Neural Network Lexical

Translation Model (NNLTM). In that period, we also built a broader variety of rankers, developed techniques for score-based late fusion and query-specific cutoff selection, and began experimenting with short-duration evaluation sprints. Principal research foci in OP2 included development of neural ranking and document expansion techniques and investigation of architectures for coupling early fusion, late fusion, and neural reranking. In the NCTE, we focused on extending our approaches to document enhancement and late fusion.

2.5 Summarization

In the BP, the novel setting and absence of training data meant that our summarization work focused on using unsupervised methods derived from word embeddings to select relevant sentences from documents. During OP1, we began to overcome the training data limitation by generating synthetic training data and using it to train supervised query-sentence relevance models to perform the same task. During OP2, we augmented our approaches using retranslation to help ensure that in cases where we are highly confident that a given document is relevant to a query, the query itself appears in the document verbatim. This reflected findings from our early work in BP that human end-users often failed to mark a summary as relevant unless the precise query terms were present—and visually highlighted—within the summary text presented. In the NCTE, we focused on the use of data augmentation techniques to develop a supervised approach to cross-lingual query-focused abstractive summarization. In addition, we worked on a paper on our work to improve relevance of summaries through constrained translation and automated post-editing (APE).

3.0 TECHNICAL APPROACHES

3.1 Speech Processing

This section describes the approaches investigated by the SCRIPTS team during the NCTE to process the audio documents so that they can be processed for MT and CLIR.

Obtaining accurate confidence scores is a key challenge of using speech recognition systems in both low-resource scenarios and across mismatched topic domains. In general, the baseline approach for obtaining confidence scores is to combine the arc-posteriors from a word lattice with the posteriors from arcs associated with the same word and time instance, in order to generate a word-level posterior. Because this approach often overestimates word-level confidence scores, in Kaldi-based systems they are typically calibrated using decision trees. During BP and OP1, we investigated alternative approaches for generating more accurate confidence scores based on deep-learning (DL) approaches [1, 2, 3], specifically lattice recurrent neural networks (RNN) and attention mechanisms.

Work continued during the NCTE on estimating confidence scores for End-to-End (E2E) models. Directly applying lattice-based approaches to these E2E models yielded poor estimates of confidence scores because the standard decoding approaches generate shallow lattices, which have limited path diversity. To address this problem, we examined techniques using the decoding state to predict confidence, but this work has not been completed.

One of the major problems for ASR in low resource settings is the lack of data in the target language to develop a supervised approach. During the NCTE, we developed a method [4] for "deciphering" target language speech from the output of a multilingual phone recognizer. This allows a flat-start semi-supervised training procedure to be run, without the need for either an initial AM for the target language, or any paired audio and text data. Such an approach would be enormously beneficial in low resource settings. We have investigated this technique on Tagalog data from MATERIAL.

3.2 MT

The key steps in developing MT systems include the data selection and cleaning process, applying SMT by generating translations through the use of statistical models that analyze features of bilingual texts, and a variety of E2E NMT approaches.

In the NCTE period, we experimented with improvements on our basic methods. First, we experimented with developing a fast Farsi→English translation system using the recipe developed for English→German translation [5] as part of the Edinburgh NMT approach. Second, we also experimented with the use of constrained MT to help in summarization. This was a cross-team effort within SCRIPTS.

After the evaluation period, we devised a controlled comparison of the various techniques for constraining translation within the context developed by the SCRIPTS team. While most prior work evaluates constrained MT in artificial settings (including the one discussed above), the MATERIAL program provides a framework for evaluating these methods with respect to human comprehension in the context of CLIR. Specifically, we can measure the impact of retranslating documents with query words as constraints to produce summaries and evaluate whether summaries based on retranslation improve human relevance judgments. The systems compared include Edit-Based Transformer with Repositing (EDITOR), the query-guided translation approach, and APE.

3.3 Text Processing

The focus of the text processing portion of the research is on developing morphological analyzers. Over the full course of the IARPA MATERIAL program, we have developed two main capabilities: 1) morphological segmentation; 2) and POS tagging. During the NCTE, our focus was on improvements to unsupervised POS tagging.

3.3.1 POS Tagging via Cross-Lingual Projection

Unsupervised cross-lingual POS tagging via annotation projection uses available POS tags from a source language to project onto a target language using word-level alignments. The projected tags then form the basis for training a POS model for the target language. The overall pipeline is illustrated in Figure 1, with the left-hand illustration outlining the word-based alignment process and the right-hand illustration showing the stem-based alignment process.

We have developed a robust approach for standardizing the process of annotation projection by exploiting and expanding upon current best practices, in order to produce reliable annotations that improve the quality of the training data for the target language POS tagger. Our approach includes: 1) using bidirectional alignments; 2) coupling token and type constraints on the target side; 3) scoring the annotated sentences for the selection of reliable training instances and 4) using both word-based and stem-based alignments.

3.3.1.1 Morpheme-Level Alignment and Projection

During the NCTE, we also examined the use of morpheme-level alignment and projection, similar to the stem-based approach. This approach abstracts away from whether the morphemes in the source and target languages are free- standing or not. We hypothesize that morpheme-based alignment and projection would help when both the source and target languages are morphologically complex, and therefore we examine this approach using Arabic as the source.

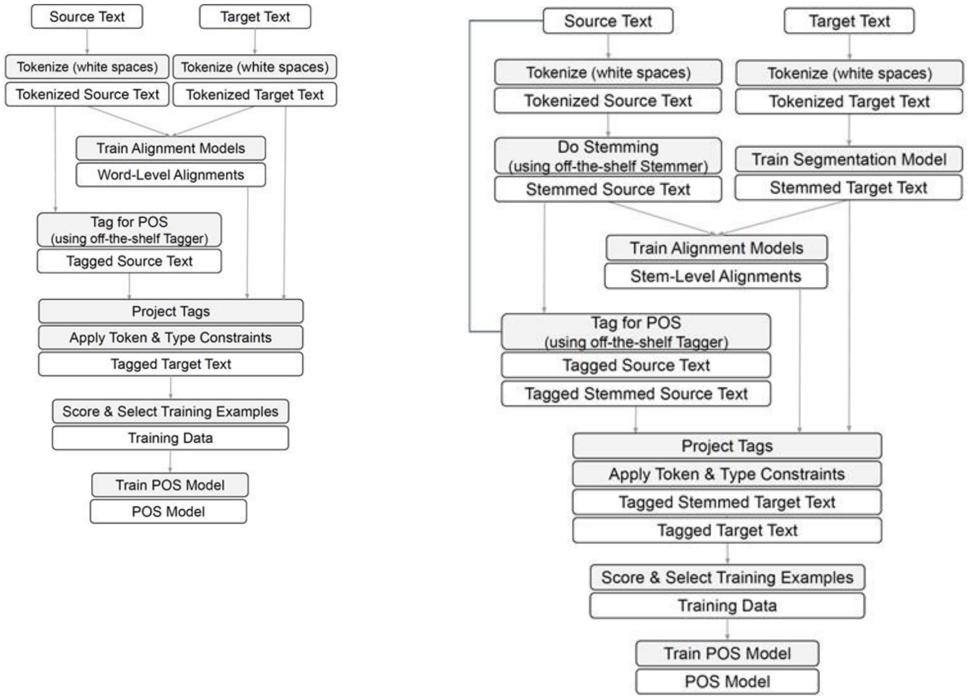
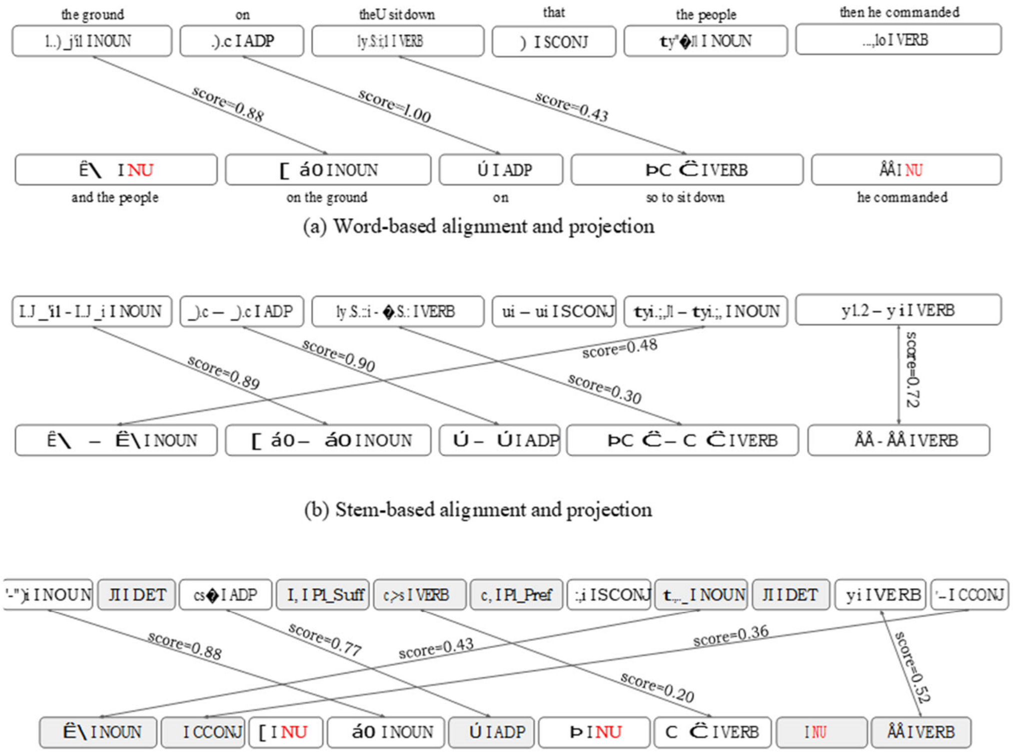


Figure 1. The Overall Pipeline for Unsupervised Cross-Lingual POS Tagging via Alignment and Projection: Word-Level Alignment (left); Stem-Level Alignment (right)

In Figure 2(c) below, it shows that conducting both alignment and projection on the morpheme level for Arabic and Amharic results in a complete POS assignment on the Amharic side as at least one morpheme from each word receives a POS tag. In this case, every sequence of consecutive morphemes in the same color corresponds to one word. The pipeline of the morpheme-based approach is similar to the one of the stem-based approach depicted in Figure 1. On the source side, each morpheme receives a separate POS tag using an off-the-shelf POS tagger; we use MADAMIRA to obtain the morphemes for Arabic. We obtain the target morphemes in inflected forms (morphs) using MorphAGram by applying the same cascaded setting we use in the stem-based approach. We then project the POS tags from the source morphemes onto the target ones through bidirectional morpheme-level alignments that are induced by morpheme-level alignment models trained in the morpheme space. Since we train the POS model on the word level, once we apply the token and type constraints, we replace each sequence of morphemes that corresponds to one word on the target side with its corresponding word and assign that word the POS tag of the representative morpheme. We define the representative morpheme either as the morpheme whose POS tag ranks the highest among those of the other morph (RANK) or as the stem morpheme (STEM). For ranking, we use the default ranking of POS tags defined at https://github.com/coastalcph/ud-conversion-tools (i.e., VERB, NOUN, PROPN, PRON, ADJ, NUM, ADV, INTJ, AUX, ADP, DET, PART, CONJ, SCONJ, X and PUNCT). Ideally, in future work, this ranking should be adjusted to be language specific. For instance, the first Amharic word in Figure 2(c) receives the NOUN tag either because NOUN supersedes CCONJ (RANK) or because NOUN is assigned to the stem morpheme (STEM).



(a) Word-based alignment and projection

(b) Stem-based alignment and projection

(c) Morpheme-based alignment and projection

3.4 CLIR

CLIR combines a set of sequentially run components, each of which carries out a number of steps in order to support search across foreign-language documents of various formats. First, indexing is performed on the following types of source documents: text documents in their original ("foreign") language, ASR transcripts of spoken foreign content, and alternative foreign words and phrases recognized by KWS (indexing). English indexes for both stems and words are also created using translation results for 1-best MT of text and speech, and translation probabilities are also stored for use with PSQ and the NN-LTM (translation). Both queries and documents are further enhanced using some combination of stemming, stopword removal, and query or document expansion, and those results are also indexed (enhancement). Query processing is also performed as a pre-processing step (query processing). The results of these steps are then used as input to several ranking methods to rank order documents (ranking). Different choices across these components result in diverse ranked lists of documents; these can then be combined into a single ranked list (late fusion). During the BP (but not during OP1 or OP2), domain and language filtering were then applied (filtering). Finally, tuned cutoffs are applied to identify the most highly-ranked documents, which are returned as the result set (cutoffs).

In the NCTE, our focus was on document enhancement and late fusion. Here, we describe the approaches to these tasks that we developed over the course of the IARPA MATERIAL program and the extensions we made during the NCTE.

3.4.1 Enhancement

Query and document representations can be enhanced in several ways before performing retrieval. Our interim report described techniques for stopword removal, stemming query expansion and document expansion. In the NCTE period we focused on Deep Contextualized Term Weighting framework (DeepCT), a document expansion technique.

3.4.2 DeepCT

Like Doc2query, DeepCT [6] is a state-of-the-art (SOTA) model added in OP2 that alters the document representation prior to BM25 ranking; in this case, however, the BERT model predicts the optimal frequency of each term in each document, and then alters the index to use that frequency. This is done by creating a new document in which each word is repeated the optimal number of times. Unlike Doc2query, DeepCT doesn't expand the document by any words which are not in the original document. These newly built documents are then indexed and used in a search.

3.4.3 Late Fusion

Combining different ranking methods, translations, query versions, and/or enhancement techniques, helps generate diverse lists of ranked documents. Using late fusion (i.e., fusion performed after ranking), allows us to benefit from this diversity. For each query, the System Combination module receives a set of lists of ranked documents on the input, along with scores for each document, and produces a single ranked list of documents. We have experimented over the course of the program with a range of methods for this purpose, some of which use only the ranks of the returned documents, and others of which also used the score associated with each document. A detailed description of these methods (apart from Hierarchical Reciprocal Rank Fusion[H-RRF]) can be found in [7].

Over the course of the IARPA MATERIAL program, we experimented with rank-based fusion, unweighted score-based fusion and weighted CombNMZ. During the NCTE, we experimented with a new method, H-RRF. We describe all methods here so that differences between methods are clear.

3.4.3.1 Rank-Based Fusion.

We used two approaches to rank-based fusion, one based on Borda counts [8] and the other based on reciprocal ranks. Borda-count fusion uses positional voting to assign a score to each document returned by the ranker as the number of returned documents - rank of the returned document. The scores for the document are then summed over all the rankers. In RRF [9], the scores are calculated as 1 / rank of the returned document. The trec_tools [10] implementation of the RRF uses an adjusted ratio of 1 / 60 + rank of the returned document which removes a disadvantage of the documents with very low ranks. Again, the scores are then summed over all the rankers.

3.4.3.2 Unweighted Score-Based Fusion

We used two approaches to unweighted score-based fusion, CombSUM and CombMNZ. The CombSUM method simply takes the scores of the document retrieved by all the rankers which are combined. CombMNZ is a refinement of this approach in which the summed score is then multiplied by the number of systems, ensuring that the documents returned by more systems are promoted. In either case, the scores from different systems can be in very different ranges and so need to be normalized and put into a shared range. For this we have used sum-to-one normalization: the scores of all documents returned by particular systems were summed, and the score for a particular document is then divided by the sum.

3.4.3.3 Weighted CombMNZ

Weighted CombMNZ further modifies CombMNZ by assigning a weight to each system—the final score is then calculated as a weighted linear combination of the scores. We tuned the weights on development data, using the Maximum Query Weighted Value (MQWV) of each system on the DEV or DEV+ANALYSIS collection as its weight.

3.4.3.4 H-RRF

In contrast to other methods for system fusion which might select a small subset of systems to combine, this method combines all the systems. The basic strategy is to combine subsets of systems together using RRF, and then later combine the results created in the first stage. In Bendersky et al. [11], the authors defined the initial groups for combination as the variations on their main systems generated by adjusting hyper-parameters. We explored two alternatives to the grouping described in that work. In one approach, we defined groups using a clustering algorithm based on the Jaccard similarity of each system's retrieved documents. Here, two systems are considered more similar if their sets of relevant documents returned are similar, and the most similar systems are then combined in the first RRF stage. Our second approach was to group systems by some aspect of their design, such as which type of machine translation technique or which type of information retrieval technique they used. Of these, we got the best results from grouping by information retrieval technique.

3.5 Summarization

The role of summarization in this work is to create a short, compelling summary of a document selected by the upstream systems (i.e., CLIR) and assist an end user in determining quickly whether it is relevant to their given query. Ideally, the document is relevant, and the summary's content will focus on the query; however, the summarizer must faithfully represent both relevant and irrelevant documents. We use extractive summarization, which generates a summary by selecting sentences verbatim from the input document, as opposed to generating new sentences.

Over the course of the IARPA MATERIAL project, we developed an unsupervised approach to summarization and a supervised approach using data augmentation. We also developed a method for retranslating or post-editing summaries in order to better reveal relevance. In the NCTE, we completed work on an approach to query-focused abstractive cross-lingual summarization that extended our earlier work on generic cross-lingual summarization. We also wrote a paper on our work on re-translation and post-editing to improve cross-lingual query-focused summarization that we plan to submit to North American Association for Computational Linguistics (NAACL).

3.5.1 Abstractive Cross-Lingual Summarization

Throughout this work, we explored a number of new approaches that were not adopted in the final evaluation system. For example, we observed that the fluency of a summary seemed to affect whether or not it was selected by Amazon Mechanical Turk (AMT) workers and reasoned that we might be able to improve the fluency of the summary translation using abstractive summarization. Because the summary is not the entire document, an abstractive approach could feasibly reword only those sentences needed for fluency. By contrast, an extractive method that would need to make a trade-off between conciseness and fluency. We first published our approach to generic abstractive cross-lingual summarization in the 2019 Annual Conference of the NAACL 2019 [12] and have since extended that work for the task of query-focused abstractive summarization.

3.5.2 Cross-Lingual Query-Focused Abstractive Summarization

Extractive summarization systems can sometimes have fluency errors, especially in the case of cross-lingual summarization where the extracted excerpts come from potentially noisy translations of the original document. Prior work has shown the humans tend to place emphasis on the fluency of an output document, even at the expense of adequacy [13]. Therefore, we developed an approach for generating query-relevant, fluent, abstractive cross-lingual summaries using the translate- then-summarize paradigm. We first devised a method to create a synthetic corpus for query-focused summarization from an existing English summarization corpus Convolutional Neural Network/ Time-Delay Neural Network Factorization (CNN/DM), using a keyword/key phrase-based query generation algorithm. We then performed round-trip translation of the original English document to get a synthetic cross-lingual corpus. We also enhance the BART model

[14] with a query focus mechanism in the encoder, which leads to SOTA performance on both our synthetic corpus, as well as better out of distribution generalization as compared to prior work. Crucially for MATERIAL, our approach allows the summarizer to correct for existing disfluencies in the translation and include the query, without requiring an additional retranslation step. Our results can be found in Section 4.5.1.

4.0 PROGRAM RESULTS, FINDINGS AND TECHNICAL INSIGHTS

4.1 Speech Processing

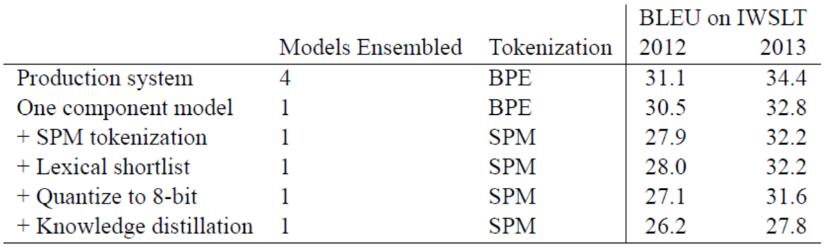
We investigated the use of a “deciphering” method to decode Tagalog narrow band (NB) and wide band (WB) data from the Analysis set. Preliminary experiments yielded a phone error rate from decipherment of 68.4%. Our experience from other languages is that this may be sufficient to achieve further gains from semi-supervised training.

4.2 MT

4.2.1 Results and Findings: Edinburgh MT (EDINMT)

We attempted to make a fast Farsi→English translation system using the recipe developed for English→German translation [5], however, there was a dramatic drop in translation quality, as shown in Table 1. We investigated these results by examining each step one at a time. SentencePiece Model (SPM) tokenization, which allows for convenient implementation in C++, de- creased quality compared to byte-pair encoding (BPE) tokenization; knowledge distillation also produced unexpected decreases in translation quality. Having identified these limitations, in future work we will broaden the set of languages tested in order to develop a general recipe, rather than one that is overfitted to the data condition for German.

Table 1. Investigation of Quality Drop when Attempting to Make a Fast Fa→En Translation System Using a Recipe Developed for High-Resource En→De Translation.
Tokenization is SPM or BPE.



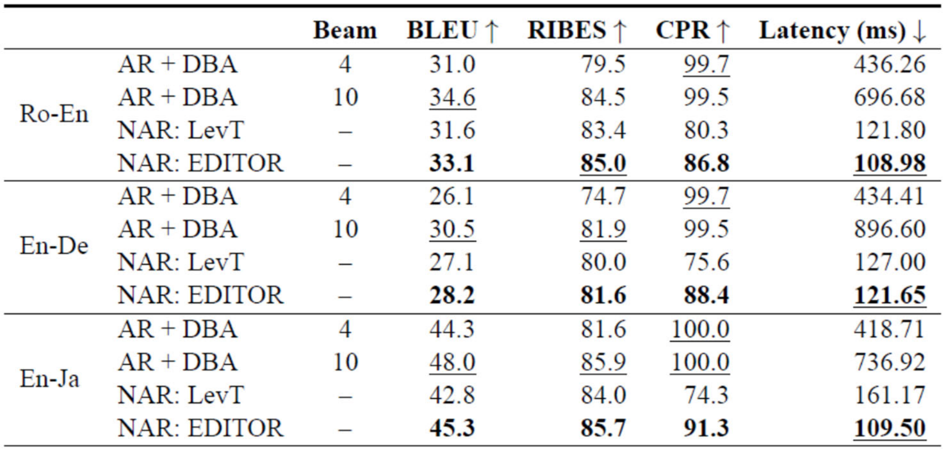
	Models Ensembled	Tokenization	BLEU on IWSLT 2012	2013
Production system	4	BPE	31.1	34.4
One component model	1	BPE	30.5	32.8
+ SPM tokenization	1	SPM	27.9	32.2
+ Lexical shortlist	1	SPM	28.0	32.2
+ Quantize to 8-bit	1	SPM	27.1	31.6
+ Knowledge distillation	1	SPM	26.2	27.8

4.2.2 Results and Findings: UMD NMT

Experimental Results for Constrained Translation Although it was not incorporated into the evaluation system, we assess EDITOR’s ability to incorporate lexical constraints into its outputs. This is motivated by the need to encourage MT systems to use terms that are consistent with the query when translating documents that have been found to be relevant. We evaluate on three translation tasks: Romanian-English, English-German, and English- Japanese translation with provided terminology constraints [15]. As shown in Table 2, we compare EDITOR with an autoregressive (AR) NMT model with constrained beam search (AR+Dynamic Beam Allocation [DBA]) [16] and Levenshtein Transformer (LevT) [17], which is the SOTA non-autoregressive (NAR) Neural Machine Translation (NMT) model. For each metric, we underline the top scores among all models and boldface the top scores among NAR models based on the paired bootstrap test with $p < 0.05$ [18].

EDITOR decodes 6–7% faster than LevT on RO-EN and EN-DE, and 33% faster on EN-JA, while achieving comparable or higher Bilingual Evaluation Understudy Score (BLEU) and Rank-based Intuitive Bilingual Evaluation Score (RIBES). As in [17], we evaluate translation quality via case-sensitive tokenized BLEU and RIBES [19], which is more sensitive to word order differences. For lexically-constrained decoding, we report the Constraint Preservation Rate (CPR) in the translation outputs. We quantify decoding speed using latency per sentence. This is computed as the average time[1] (in milliseconds) required to translate the test set using a batch size of one divided by the number of sentences in the test set.

Table 2. Performance of EDITOR Compared with AR And NAR Translation Baselines on Lexically Constrained MT.



		Beam	BLEU ↑	RIBES ↑	CPR ↑	Latency (ms) ↓
Ro-En	AR + DBA	4	31.0	79.5	99.7	436.26
	AR + DBA	10	34.6	84.5	99.5	696.68
	NAR: LevT	–	31.6	83.4	80.3	121.80
	NAR: EDITOR	–	**33.1**	**85.0**	**86.8**	**108.98**
En-De	AR + DBA	4	26.1	74.7	99.7	434.41
	AR + DBA	10	30.5	81.9	99.5	896.60
	NAR: LevT	–	27.1	80.0	75.6	127.00
	NAR: EDITOR	–	**28.2**	**81.6**	**88.4**	**121.65**
En-Ja	AR + DBA	4	44.3	81.6	100.0	418.71
	AR + DBA	10	48.0	85.9	100.0	736.92
	NAR: LevT	–	42.8	84.0	74.3	161.17
	NAR: EDITOR	–	**45.3**	**85.7**	**91.3**	**109.50**

These results show that EDITOR exploits soft lexical constraints more effectively than the LevT while also speeding up decoding as compared to the constrained beam search implementation used.

We are continuing our controlled comparison of the various techniques for constraining translation within the context developed by the SCRIPTS team, including EDITOR, query-guided translation and APE, and plan to submit the results for review in 2022 at a top-tier conference venue such as NAACL or Empirical Methods in Natural Language Processing (EMNLP).

4.3 Text Processing

In this section, we present the results and findings obtained during the NCTE for the unsupervised POS tagging portions of this research.

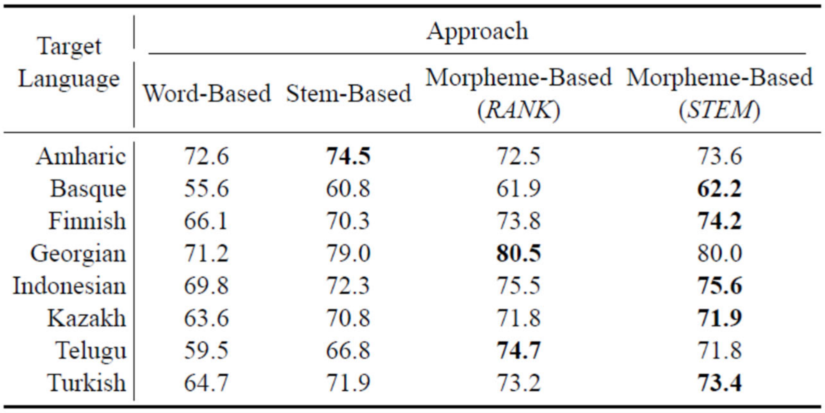
4.3.1 POS Tagging (Neural Model): Results

We have conducted most of our experimentation with the Neural POS tagger, as that was superior to the Average Perceptron Model in our earlier experiments.

Morpheme-Based Alignment Experiments We evaluate the morpheme-based approach when projecting from Arabic, our source language of the richest morphology, using the RANK and STEM mechanisms for the selection of the representative morphemes. We compare the results to those of the word-based and STEM-based approaches and report them in Table 3. The best result per target language is in bold.

[1] Excluding the model loading time.

Table 3. The POS-Tagging Performance (Accuracy) of the Word-Based, STEM-Based and Morpheme-Based Approaches when Projecting from Arabic using the New Testament as the Source of Parallel Data.



Target Language	Approach			
	Word-Based	Stem-Based	Morpheme-Based (*RANK*)	Morpheme-Based (*STEM*)
Amharic	72.6	**74.5**	72.5	73.6
Basque	55.6	60.8	61.9	**62.2**
Finnish	66.1	70.3	73.8	**74.2**
Georgian	71.2	79.0	**80.5**	80.0
Indonesian	69.8	72.3	75.5	**75.6**
Kazakh	63.6	70.8	71.8	**71.9**
Telugu	59.5	66.8	**74.7**	71.8
Turkish	64.7	71.9	73.2	**73.4**

The morpheme-based approach results in denser training instances, as both alignment and projection are performed at a more fine-grained level than those in the word-based and STEM-based approaches. With the exception of Amharic, the morpheme-based approach yields the best performance for all the target languages when projecting from Arabic. Telugu benefits the most, with relative error reductions of 23.9% and 15.3% over the stem-based approach using the RANK and STEM mechanisms, respectively. The difference in the performance of the RANK and STEM mechanisms is only statistically significant for p-value < 0.01 in Amharic and Basque, where the STEM mechanism yields better performance, and in Telugu, where the RANK mechanism gives a better result. We observe that the quality of morphological segmentation may affect the detection of stems, and thus might affect the quality of the STEM mechanism. However, all the improvements achieved using the morpheme-based approach—rather than the STEM-based one—are statistically significant.

4.4 CLIR

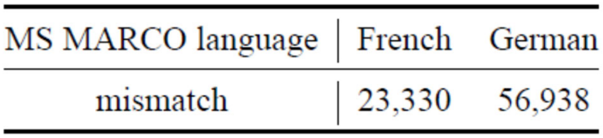
In this section, we present the results and findings obtained during the NCTE for the CLIR portions of this research.

4.4.1 Document Expansion

To experiment with DeepCT document expansion, we translated the Microsoft Machine Reading Comprehension (MS MARCO[2]) [20] triples (each containing a query, a positive example, and a negative example) to the document language and trained a DeepCT model initialized using a multilingual BERT (mBERT) encoder. We call this approach Translate-Train (TT). As Table 4 shows, translation divergences cause this approach to fail for at least one of the two passages in 5% to 10% of the 500k translated MS MARCO training passages.

[2] https://microsoft.github.io/msmarco/

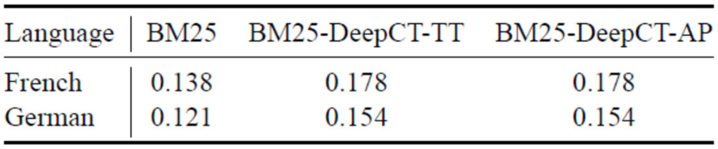
Table 4. MS MARCO Training Triples with No Lexical Match Between a Translated Query and the Translated Positive or Negative Example Passage.



MS MARCO language	French	German
mismatch	23,330	56,938

In such cases, we instead tried to align the original English query term (before translation) to the most probable translation of that English term that actually is present in the translated passage. We call this approach Annotation Projection (AP). Table 5 shows Mean Reciprocal Rank (MRR @10) results for monolingual experiments on MTs of the MS MARCO DEV set into French or German. As a baseline, we report results for “vanilla” BM25 (i.e., without DeepCT). Employing BM25 with the TT variant of DeepCT (BM25-DeepCT-TT) outperforms vanilla BM25, demonstrating that contextualized multilingual embeddings can generate useful term weights. However, also incorporating annotation projection in the no-match cases (BM25-DeepCT-AP) yields no further improvement in MRR@10. We view these results as suggestive rather than confirmatory because the evaluation collection is synthetic (itself created using MT) and because the experiments are monolingual. Still, the results do indeed suggest that training DeepCT for languages other than English can be done using our TT approach.

Table 5. MRR@10 on MTs of the MS MARCO DEV Set.

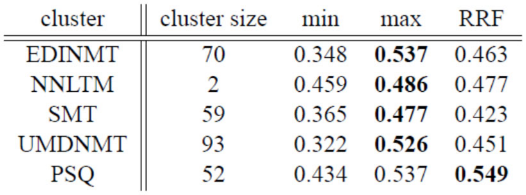


Language	BM25	BM25-DeepCT-TT	BM25-DeepCT-AP
French	0.138	0.178	0.178
German	0.121	0.154	0.154

4.4.2 Late Fusion

Combining several individual systems to improve performance has been successfully used in different research areas and applications, including ML [21], speech recognition [22], and information retrieval (IR) [7]. For late fusion (i.e., combination of ranked lists) to be helpful, the combined systems need to be both well-performing (although not necessarily best-performing) and diverse. Late fusion proved beneficial in almost all of our experiments, but in building a large number of configurations (for some languages, more than 1,000 different setups) we encountered new research problems. For example, which systems should be combined to achieve the best performance? We presented an assessment of different methods in our interim report. Here, we present new results obtained during the NCTE.

Table 6. Mean Average Precision (MAP) for Clustering by MT Technique, Georgian DEV Text.



cluster	cluster size	min	max	RRF
EDINMT	70	0.348	**0.537**	0.463
NNLTM	2	0.459	**0.486**	0.477
SMT	59	0.365	**0.477**	0.423
UMDNMT	93	0.322	**0.526**	0.451
PSQ	52	0.434	0.537	**0.549**

Table 7. MAP for Clustering by IR Technique, Georgian DEV Text.

cluster	cluster size	min	max	RRF
anserini	21	0.370	**0.484**	0.477
hmm	30	0.491	0.537	**0.543**
indri	203	0.322	**0.473**	0.452
neulex	2	0.459	**0.486**	0.477
psqinter	19	0.496	**0.537**	**0.537**
seclr	1	0.457	**0.457**	**0.457**

Table 8. MAP for RRF of Results from Each Cluster, Georgian DEV Text.

	MT Clusters	IR Clusters	Jaccard Clusters
Hierarchical RRF combination	0.515	**0.587**	0.462
RRF combination of best from each cluster	0.555	**0.596**	0.507

After the evaluation period, we experimented with hierarchical combinations of all systems (Section 3.4.2), focusing on the grouping systems at the first stage. We first grouped systems by the MT technique they use (e.g., all systems using SMT were clustered). This resulted in five clusters, the size of which varied considerably. Table 6 shows the best and worst MAP for the systems in each cluster, along with the mean average precision (MAP) for a RRF combination of the systems in each cluster. As can be seen, RRF often yields results between the best and worst MAP for individual systems. Alternatively, we can group systems by the IR technique that they use. As Table 7 shows, RRF over the systems in each cluster also often yields MAP no better than the MAP of the best system in that cluster. Our principal focus, however, is on hierarchical combination, in which we perform a second RRF over the RRF results from each cluster. As Table 8 shows, when used with IR clusters, this hierarchical combination approach yields MAP values better than any individual system, and better than the one-stage RRF of any of the sets of systems shown. As the second row of that table shows, using RRF over only the best system in each cluster would have done even better, but that oracle baseline cannot be built in practice because the best systems on the actual test data are not known a priori. For reference, we also include an unsupervised clustering approach in Table 8, where clusters are formed using the Jaccard similarity of the documents found by each system; this does not work as well. From this we conclude that hierarchical RRF combination using clustering by IR technique is a promising approach.

Across all tables, the bolded values are the best per row.

4.5 Summarization

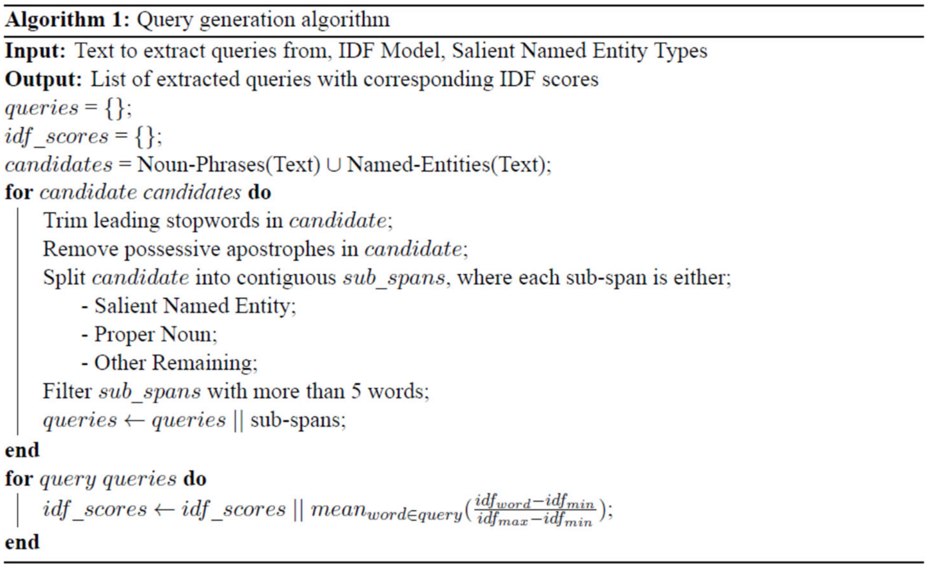
4.5.1 Use of Constrained Translation or APE

We have written a paper, to be submitted, on our experiments with retranslation and are able to demonstrate the effectiveness of this approach on lexical queries specifically.

4.5.2 Cross-lingual Query-Focused Abstractive Summarization

To generate training data for our task, we needed a parallel corpus with a foreign language and a poor translation, such as would be produced by MT, with the input document in the foreign language and the summary in translated English. The corpus was generated by using opensource Opus MT models to create the round-trip translated version of the CNN-Daily Mail corpus [23, 24], using Arabic as the LRL. The query generation method used to create the query-focused corpus is described in Algorithm 1.

We add a query focus mechanism to the BART [25] encoder by explicitly augmenting the embeddings of the tokens in the document that correspond to the query; these focus embeddings are initialized randomly and learned during the finetuning stage for query-focused summarization. Results comparing this model to prior work on query-focused summarization on the query-focused summarization (QFS) corpus can be found in Table 9. We also asked humans to rate the summaries produced by each of the systems for fluency, relevance (how relevant the summaries are to the specified query), and coverage (completeness of the summary content with respect to the query). Our proposed system outperformed the baseline across all three aspects, as shown in Table 10.



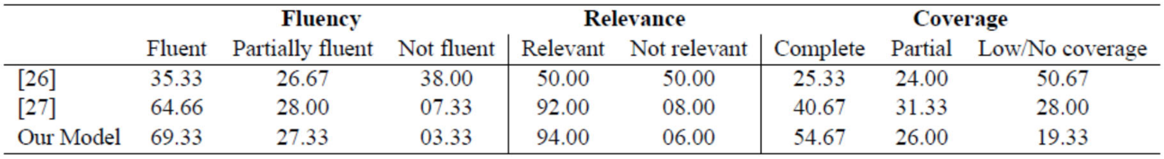
```
Algorithm 1: Query generation algorithm
Input: Text to extract queries from, IDF Model, Salient Named Entity Types
Output: List of extracted queries with corresponding IDF scores
$queries$ = {};
$idf\_scores$ = {};
$candidates$ = Noun-Phrases(Text) ∪ Named-Entities(Text);
for $candidate$ $candidates$ do
    Trim leading stopwords in $candidate$;
    Remove possessive apostrophes in $candidate$;
    Split $candidate$ into contiguous $sub\_spans$, where each sub-span is either;
        - Salient Named Entity;
        - Proper Noun;
        - Other Remaining;
    Filter $sub\_spans$ with more than 5 words;
    $queries \leftarrow queries$ || sub-spans;
end
for $query$ $queries$ do
    $idf\_scores \leftarrow idf\_scores \,||\, mean_{word \in query}(\frac{idf_{word}-idf_{min}}{idf_{max}-idf_{min}})$;
end
```

Figure 3. Algorithm 1

Table 9. Automatic Summarization Metrics Comparing our Abstractive Query Focused Model to Baselines on the Generated Query-Focused Summarization (QFS) Corpus.

	ROUGE 1	ROUGE 2	ROUGE L
[26]	13.01	2.66	12.13
BART [14]	23.97	7.78	20.36
BART with Constrained Decoding [27]	25.78	8.74	21.42
Our Model	37.61	19.09	33.12

Table 10. Human Evaluation Results - Accuracy of Fluency, Relevance and Coverage as Annotated by Human Judges.



	Fluency			Relevance		Coverage		
	Fluent	Partially fluent	Not fluent	Relevant	Not relevant	Complete	Partial	Low/No coverage
[26]	35.33	26.67	38.00	50.00	50.00	25.33	24.00	50.67
[27]	64.66	28.00	07.33	92.00	08.00	40.67	31.33	28.00
Our Model	69.33	27.33	03.33	94.00	06.00	54.67	26.00	19.33

We also perform an ablation to understand the impact of focus embeddings and find that both the query prefix (allowing for query focusing through self-attention), and focus embeddings result in a significant increase in performance. Results are shown in Table 11.

Table 11. Ablation Study of our Abstractive QFM- Automatic Summarization Metrics on our Corpus to Evaluate the Impact of Focus Embeddings.

	ROUGE 1	ROUGE 2	ROUGE L
Without Query Prefix and Focus Embeddings	23.97	7.78	20.36
Only Query Prefix	36.06	17.80	31.82
Query Prefix and Focus Embeddings	37.61	19.09	33.12

5.0 CONCLUSIONS AND RECOMMENDATIONS

Here we present the reflections and recommendations of each team whose work was part of the SCRIPTS development process.

5.1 Speech Processing

The SCRIPTS ASR research findings highlighted that porting systems across domains is possible even for low-resource languages, as long as sufficient data for the language and approximate domain is available. Notably, however, we find that even un0transcribed audio data can fill this data role, which helps transform the ASR possibilities for low-resource languages thanks to the ability to scrape audio from the web. At the same time, it is still important to address the impact of errors from the ASR system when using its outputs with downstream components.

A key area of future investigation for ASR on LRLs is to explore the degree of domain task porting that is possible when the availability of either text or audio data in the target language is limited. Our work indicates that the use of large volumes of un-transcribed audio data is essential for ASR system development in the LRL context. Yet while unsupervised pre-training with such systems is a starting point for this approach, additional research is required to improve this type of ASR system development. Likewise, exploring how to link these large pre-trained systems with the particular needs of downstream systems will help yield more robust results.

5.2 MT

The MT work in this project reinforced the importance of robust data collection methods and pre-processing pipelines in order to quickly develop systems for new language pairs. It also demonstrated that neural models can provide useful translations even in LRL settings, as well as the benefits of tailoring MT systems to the distinct needs of upstream and downstream components. In the MATERIAL program, we confirmed this by improving results for other SCRIPTS component systems through techniques including dedicated data processing, providing alter- native translation options, and incorporating query-specific constraints in translation.

A key area for future exploration is how to effectively measure the impact of MT errors on downstream components and users, which cannot always be reliably estimated at present. Investigating how to support closer cooperation and tighter integration between MT and other components—even with minimal E2E training data available—would also enhance the efficacy and applicability of future MT work in this area.

5.3 Text Processing

The results of the SCRIPTS morphological analysis work illustrate that unsupervised approaches - especially when guided by some form of linguistic priors—can obtain good performance for LRLs of diverse linguistic typology, from agglutinative to polysynthetic. For morphological segmentation, the use of AGs allows the incorporation of linguistic priors in the form of either grammar definition or linguist-provided affixes. Moreover, for POS tagging, learning from multiple source languages via either decoding or projection had a positive impact. We also found that considering the stem as the unit of abstraction for alignments is powerful - particularly for morphologically complex languages.

A primary recommendation for future work in this area is to develop mechanisms for more robustly evaluating the impact of morphological analysis on downstream tasks like MT.

5.4 CLIR

From the perspective of CLIR, the timing of the MATERIAL program was exquisite. Prior to MATERIAL, a solid foundation of "traditional" approaches to CLIR—approaches built on sparse term representations - had already been developed. The concurrent turn toward neural DL across speech recognition, MT, and IR made the MATERIAL program an unprecedented opportunity to advance not just those individual technologies, but their integration and application to the CLIR task.

For CLIR, our most important recommendation would be to consider how best to capture and share the training data required by data hungry neural methods. While such training data need not be fully integrated across speech, translation, and retrieval, given the outsize impact that training data has on these systems' effectiveness, some consideration of those requirements will be both possible and necessary. Our work on the MATERIAL program leads us to suspect that there will be benefits to thinking of these data needs as closely coupled, just like the resulting systems. As a simple example, training information retrieval (IR) systems can benefit from active learning; but the benefit for CLIR might be that much greater if the learning were to be performed on parallel rather than monolingual sources.

5.5 Summarization

Summarization research showed that an effective summarization model can be developed even when there is a total absence of annotated data for the query-focused task, thanks to synthetic data generation. Our work in the MATERIAL program demonstrated the benefit of synthetic data generation, especially when combined with rationale training for lexical queries; we therefore extended this approach to conceptual queries. We were also able to develop abstractive cross-lingual summarization models by adapting our generation of synthetic data, through a process of adding "noise" to English input articles from an existing summarization dataset using backtranslation. We also showed how the generation of effective summaries in the LRL scenario depends on close integration with both ASR and MT.

Our primary recommendation going forward is to explore new task and evaluation designs that could encourage more creative solutions to summarization in order to maximize the utility of the summarization process for human end-users. Based on our findings, it is clear that abstractive summarization could play a larger role in presenting more comprehensible summaries to a human end-user, but the current evaluation framework did not reward solutions taking this approach.

6.0 REFERENCES

[1] Anton Ragni, Qiujia Li, M. J. F. Gales, and Yongqiang Wang. Confidence Estimation and Deletion Prediction Using Bidirectional Recurrent Neural Networks. In 2018 IEEE Spoken Language Technology Workshop, SLT 2018, Athens, Greece, 18-21 December 2018, pages 204–211. IEEE, 2018.

[2] Qiujia Li, Preben Ness, Anton Ragni, and Mark J. F. Gales. Bi-Directional Lattice Re-Current Neural Networks for Confidence Estimation. In IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, 12-17 May 2019, pages 6755–6759. IEEE, 2019.

[3] Alexandros Kastanos, Anton Ragni, and M. J. F. Gales. Confidence Estimation for Black Box Automatic Speech Recognition Systems Using Lattice Recurrent Neural Networks. In 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, 4-8 May 2020, pages 6329–6333. IEEE, 2020.

[4] Ondvrrej Klejch, Electra Wallington, and Peter Bell. Deciphering Speech: A Zero-Resource Approach to cross-Lingual Transfer in ASR. In arXiv:2111.06799, 2021.

[5] Maximiliana Behnke, Nikolay Bogoychev, Alham Fikri Aji, Kenneth Heafield, Graeme Nail, Qianqian Zhu, Svetlana Tchistiakova, Jelmer van der Linde, Pinzhen Chen, Sidharth Kashyap, et al. Efficient Machine Translation with Model Pruning and Quantization. In Proceedings of the Six Conference on Machine Translation. Association for Computational Linguistics, 2021.

[6] Zhuyun Dai and Jamie Callan. Context-Aware Sentence/Passage Term Importance Estimation for First Stage Retrieval. arXiv preprint arXiv:1910.10687, 2019.

[7] S. Wu. Data Fusion in Information Retrieval. Springer, 2012.

[8] JC de Borda. Mémoire sur les élections au scrutin. Histoire de l'Academie Royale des Sciences pour 1781 (Paris, 1784), 1784.

[9] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods. In Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09, page 758–759, New York, NY, USA, 2009. Association for Computing Machinery.

[10] Joao Palotti, Harrisen Scells, and Guido Zuccon. Trectools: An Open-Source Python Library for Information Retrieval Practitioners Involved in Trec-Like Campaigns. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1325–1328, 2019.

[11] Michael Bendersky, Honglei Zhuang, Ji Ma, Shuguang Han, Keith B. Hall, and Ryan T. McDonald. RRF102: Meeting the TREC-COVID Challenge with a 100+ Runs Ensemble. CoRR, abs/2010.00200, 2020.

[12] Jessica Ouyang, Boya Song, and Kathy McKeown. A Robust Abstractive System for Cross-Lingual Summarization. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 2025–2031, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[13] Marianna Martindale and Marine Carpuat. Fluency Over Adequacy: A Pilot Study in Measuring User Trust In Imperfect MT. In Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track), pages 13–25, Boston, MA, March 2018. Association for Machine Translation in the Americas.

[14] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mo- hamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.

[15] Weijia Xu and Marine Carpuat. EDITOR: An Edit-Based Transformer with Reposition- ing for Neural Machine Translation with Soft Lexical Constraints. Transactions of the Association for Computational Linguistics, 9:311–328, 03 2021.

[16] Matt Post and David Vilar. Fast Lexically Constrained Decoding with Dynamic Beam Allocation for Neural Machine Translation. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Hu- man Language Technologies, Volume 1 (Long Papers), pages 1314–1324, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[17] Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. Advances in Neural Information Processing Systems, 32:11181–11191, 2019.

[18] Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 176–181, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[19] Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. Automatic evaluation of translation quality for distant language pairs. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 944–952, Cambridge, MA, October 2010. Association for Computational Linguistics.

[20] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A Human Generated Machine Reading Comprehension Dataset. CoRR, abs/1611.09268, 2016.

[21] J. Kittler, M. Hatef, Robert Duin, and Jiri Matas. On Combining Classifiers. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 20:226–239, 01 2002.

[22] Jonathan G Fiscus. A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). In 1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings, pages 347–354. IEEE, 1997.

[23] Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching Networks for One Shot Learning. CoRR, abs/1606.04080, 2016.

[24] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. Abstractive Text Summarization using Sequence-to-Sequence RNNs and Beyond. In Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning, pages 280–290, Berlin, Germany, August 2016. Association for Computational Linguistics.

[25] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension. CoRR, abs/1910.13461, 2019.

[26] Johan Hasselqvist, Niklas Helmertz, and Mikael Kågebäck. Query-Based Abstractive

Summarization using Neural Networks. CoRR, abs/1712.06100, 2017.

[27] Yuning Mao, Xiang Ren, Heng Ji, and Jiawei Han. Constrained Abstractive Summarization: Preserving factual Consistency with Constrained Generation. CoRR, abs/2010.12723, 2020.

7.0 LIST OF ABBREVIATIONS, ACRONYMS AND SYMBOLS

AG	Adaptor Grammar
APE	Automatic Post-Editing
AP	Annotation Projection
AM	Acoustic Model
AMT	Amazon Mechanical Turk
AR	Autoregressive
ASR	Automatic Speech Recognition
BERT	Bi-directional Encoder Representations from Transformers
BLEU	Bilingual Evaluation Understudy Score
BP	Base Period
BPE	Byte-Pair Encoding
CFG	Context Free Grammar
CLIR	Cross-Lingual Information Retrieval
CNN/DM	Convolutional Neural Network/ Time-Delay Neural Network Factorization
CPR	Constraint Preservation Rate
CTS	Conversational Telephone Speech
CUED	Cambridge University
DBA	Dynamic Beam Allocation
DeepCT	Deep Contextualized Term Weighting Framework
DL	Deep Learning
DRMM	Deep Relevance Matching Model
E2E	End-To-End
EDIN	Edinburgh
EDINMT	Edinburgh Machine Translation
EDITOR	Edit-Based Transformer with Repositing
EMNLP	Empirical Methods in Natural Language Processing
H-RRF	Hierarchical Reciprocal Rank Fusion
IARPA	Intelligence Advanced Research Projects Activity
IR	Information Retrieval
KWS	Keyword Spotting
LevT	Levenshtein Transformer

LM	Language Model
LRL	Low-Resource Language
MAP	Mean Average Precision
MATERIAL	Machine Translation for English Retrieval of Information in Any Language Multilingual Bi-directional Encoder Representations from Transformers
mBERT	Multilingual BERT
MS-MARCO	Microsoft Machine Reading Comprehension
MT	Machine Translation
NAACL	North American Association for Computational Linguistics
NAR	Non-Autoregressive
NCTE	No-Cost Technical Extension
NB	Narrow Bank
NMT	Neural Machine Translation
NNLTM	Neural-Network Lexical Translation Model
OP	Option Period
POS	Part-Of-Speech
PSQ	Probabilistic Structured Queries
RIBES	Rank-Based Intuitive Bilingual Evaluation Score
RNN	Recurrent Neural Networks
SCRIPTS	System for CRoss-language Information Processing, Translation and Summarization
SPM	Sentence-Piece Model
SOTA	State-of-the-Art
TT	Translate-Train
UMD	University of Maryland
WB	Wide Band