INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Multilingual Data Selection For Low Resource Speech Recognition

Samuel Thomas, Kartik Audhkhasi, Jia Cui, Brian Kingsbury and Bhuvana Ramabhadran

IBM T.J. Watson Research Center, Yorktown Heights, USA
{sthomas,kaudhkha,jiacui,bedk,bhuvana}@us.ibm.com

Abstract

Feature representations extracted from deep neural network-based multilingual frontends provide significant improvements to speech recognition systems in low resource settings. To effectively train these frontends, we introduce a data selection technique that discovers language groups from an available set of training languages. This data selection method reduces the required amount of training data and training time by approximately 40%, with minimal performance degradation. We present speech recognition results on 7 very limited language pack (VLLP) languages from the second option period of the IARPA Babel program using multilingual features trained on up to 10 languages. The proposed multilingual features provide up to 15% relative improvement over baseline acoustic features on the VLLP languages.

Index Terms: Multilingual features, acoustic models, deep neural networks, low resource speech recognition.

1. Introduction

Although acoustic models for state-of-the-art speech recognition systems are typically trained on several hundred hours of task specific training data, in low resource scenarios only a few hours of annotated training data are often available. In these settings, it is possible to take advantage of transcribed data from other languages to build multilingual acoustic models [1, 2]. With deep neural networks (DNNs) becoming popular for acoustic modeling, several variants of these networks have been proposed for speech recognition in low resource settings [3–15]. They typically fall into the following three broad classes:

(a) Networks that use a common phoneme set covering all the languages in the training set to train a multilingual acoustic model [3, 4].

(b) Networks trained with multiple language specific output layers to alleviate the burden of finding a common multilingual phoneme set. These networks are first trained with separate output layers for each language in the training set, and then fine-tuned to the final target language [6–8].

(c) Networks trained as described above, but used to extract multilingual bottleneck features for subsequent processing instead of directly being used as acoustic models [5, 9, 10].

For training all these classes of networks, it is useful to determine the right amount of multilingual training data and the languages that contribute most to training effective acoustic models [13, 14]. In this paper, we investigate an approach to guide data selection for training multilingual feature frontends, in the spirit of the class (c) models described above. The proposed data-driven technique, which is based on an analysis of phoneme confusion matrices, allows for similarities between languages available for training to be visualized. By requiring limited amounts ($\approx$ 3 hours) of transcribed data for analysis, the method circumvents the need to transcribe large amounts of data for training. Only candidate languages from the selected language clusters now need to be transcribed for training the multilingual feature frontends. Our experiments show that frontends trained on only the selected languages can perform as well as frontends trained on the entire available data. This leads to close to 50% reduction in the amount of transcribed data and the time required for training the frontend.

The remainder of the paper is organized as follows. In section 2, we describe the multilingual feature frontend [15] used to produce multilingual representations for IBM's speech recognition and keyword search systems used in the Babel [16] Option Period 2 (OP2) evaluation. Although this multilingual frontend can be trained in advance of the evaluation period, training the model on close to 1000 hours of speech from 10 languages is time consuming. To increase the efficacy of this model, we investigate the use of multilingual data sampling. Section 3 describes the proposed data selection technique and its application to an available pool of 10 languages [17–29, 29–33]. Section 4 describes experiments and results using the multilingual frontend and the identified language clusters. The paper concludes with a discussion in section 5.

2. The Multilingual Feature Frontend

The feature frontend used in this paper employs two DNNs in a hierarchical fashion [15]. Similar to architectures proposed in [5], while the first neural network in the hierarchy is trained on acoustic features extracted from the data, the second network models intermediate multilingual representations extracted from the bottleneck layer of the first network. Both the networks are trained on data from several languages by using language-specific output layers, instead of mapping the data to a common phoneme set. The final output of this feature frontend is a multilingual representation from the bottleneck layer of the second network.

In our training framework, we use 40-dimensional log-Mel filterbank features spliced together with a context of ±5 frames as input to the first NN. The 80-dimensional bottleneck features extracted from the first network are then used as features for the second DNN. The context of these multilingual features is expanded to ±10 frames but is then subsampled at a two frame rate to produce a 400-dimension feature vector. Both DNNs use up to 10 independent output softmax layers corresponding to 10

This effort uses various language collections released by the IARPA Babel Program. Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

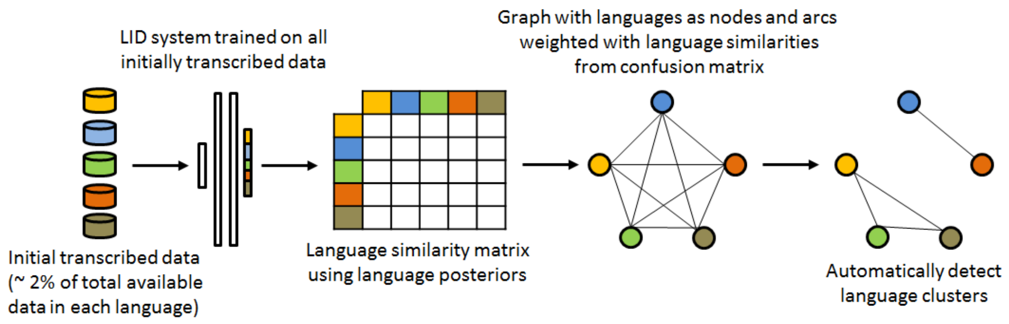

LID system trained on all initially transcribed data
Graph with languages as nodes and arcs weighted with language similarities from confusion matrix
Initial transcribed data (~ 2% of total available data in each language)
Language similarity matrix using language posteriors
Automatically detect language clusters

Figure 1: Identification of language clusters using scores from an LID system

training languages used in the Base and OP1 evaluation periods of the Babel program [9]. These languages include Assamese, Bengali, Pashto, Turkish, Tagalog, Vietnamese, Haitian Creole, Lao, Tamil and Zulu. By sharing fully connected hidden layers across all languages, while this architecture learns a multilingual representation, it also has an advantage of not requiring a common phoneme set that covers all the training languages. Using the standard error back-propagation for minimizing the cross-entropy objective function, the DNNs are trained on alignments produced by HMM-GMM acoustic models trained on each language separately.

In the context of the Babel program, training a multilingual feature frontend using 1000 hours of data across 10 training languages available is time consuming. It would hence be advantageous to train a similar-performing network on significantly fewer hours of speech. Multilingual data selection can also be beneficial in a different setting. For a new low resource language, if one had access to large amounts of untranscribed data from several other languages, it would be cost effective to know that transcribing a certain set of languages is more important than attempting to transcribe all the languages to build a multilingual frontend. In the next section we show how language clusters can be identified with up to 3 hours of transcribed data from each of the available languages. The languages falling under the dominant cluster can then be selected as candidates for transcription rather than blindly transcribing all the data.

3. Detecting Language Clusters

To detect similarities between languages and subsequently language clusters, we investigate the use of a data-driven technique based on an analysis of confusion matrices. These confusion matrices are estimated via two methods described below.

3.1. Language clusters using scores from a language identification network

In [14], to find similarities between languages, a language identification approach is proposed. This technique works by first training a shallow neural network (NN) to predict language posterior probabilities and then averaging the posterior scores over frames. For a set of languages that are used to train the language identification (LID) network, pairs of languages that are close to each other are shown to have higher predicted posteriors. We explore this technique further by training a similar LID network that discriminates between the context independent phonemes of all the languages we wish to identify. We then combine the scores corresponding to phonemes of each individual language to obtain global language scores. The combined context independent phoneme set is created by collecting context independent phonemes across all languages and keeping them language specific, i.e. without merging any phonemes although they might be acoustically similar.

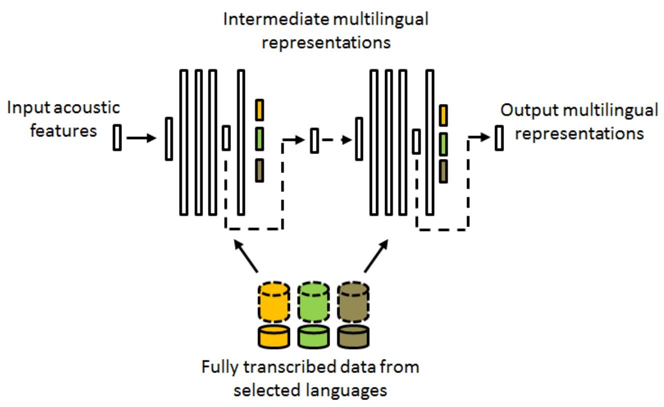
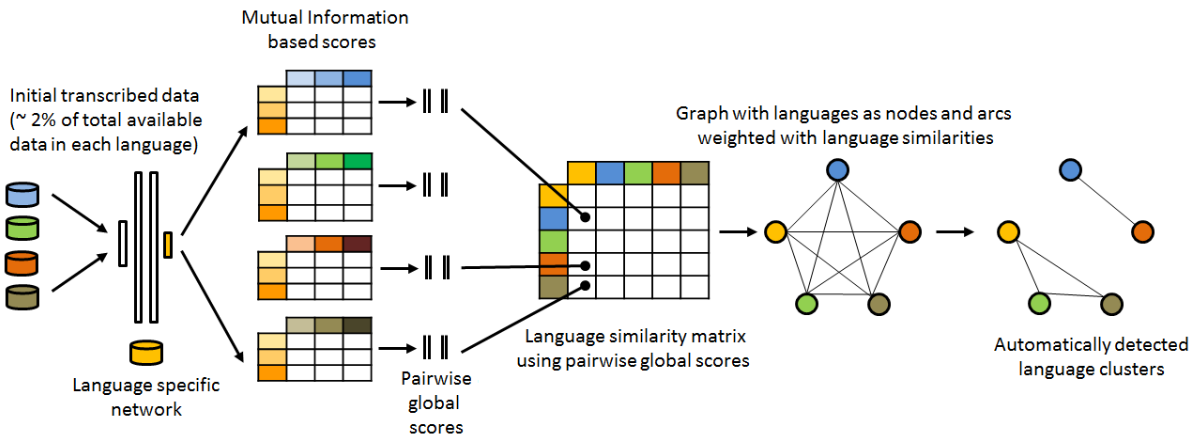

Intermediate multilingual representations
Input acoustic features
Output multilingual representations
Fully transcribed data from selected languages

Figure 2: Schematic of the hierarchical multilingual feature frontend using selected languages

As shown in Fig. 1 after an NN based LID system is trained, feature frames of each language are passed through the network to first derive posteriors of context independent phonemes of languages, before phonemes of each individual language are collapsed to a single language score. After averaging the language scores over the number of input feature frames in each language, the scores are then used to populate a language similarity matrix. The language similarity matrix is further used to construct a graph where individual nodes correspond to languages and connecting arcs are weighted by scores from the language similarity matrix. Spectral clustering is then applied to this graph to form language clusters, by solving a convex relaxation of the normalized graph cut problem [34].

To train the LID network, only a very small subset of the training data of each language is used. In our experiments we use about 3 hours of transcribed data for each language (2% of the available data). The remaining data is only used or transcribed if it belongs to a dominant language cluster, hence saving on the cost of transcribing large data sets. For the 10 Babel training languages in hand, we train an LID system on about 30 hours of speech using about 3 hours of speech from each language. A network with 3 hidden layers is trained to discriminate between 435 context independent phonemes, which are combined during test time to produce 10 dimensional language



Mutual Information based scores
Initial transcribed data (~ 2% of total available data in each language)
Graph with languages as nodes and arcs weighted with language similarities
Language specific network
Pairwise global scores
Language similarity matrix using pairwise global scores
Automatically detected language clusters

Figure 3: Identification of language clusters using scores from individually trained neural networks

posteriors. These language posteriors are then averaged over all input frames and are used to populate a language similarity matrix. After constructing a language graph and automatically partitioning it, we pick languages in the dominant cluster to train a hierarchical multilingual feature frontend as shown in Fig. 2.

By pooling together languages, although this approach can jointly discriminative between languages, we observe that as the number of the languages being compared increases, the language scores tend to spread out among classes. Empirically we notice that for fewer than 3 languages this approach can predict strong relationships. However for settings with 10 languages, the final scores across languages are often uniformly spread out, limiting the discovery of strong relationships between languages. A second limitation of this approach is that the LID system needs to be retrained every time a new language is added to the multilingual pool for selection since the LID NN is jointly trained across all languages. To alleviate these limitations, we investigate how individual language networks can be trained and combined to produce similarity scores useful for identifying language clusters.

3.2. Language clusters using scores from individual language networks

To discover language clusters from individual language networks, we begin by estimating confusion matrices between language pairs. To create a confusion matrix between two languages $\mathcal{L}_\mathcal{A}$ and $\mathcal{L}_\mathcal{B}$, with phoneme sets A and B, we train neural networks on both languages. A network trained on $\mathcal{L}_\mathcal{A}$ for example, estimates posterior probabilities of speech sounds in A, conditioned on the input feature vectors. We then forward pass the data in $\mathcal{L}_\mathcal{B}$ through the trained NN. To understand the relationship between phonemes, we treat the phoneme recognition system as a discrete, memory-less, noisy communication channel with the phonemes in B as source symbols to the system. Using the recognized phonemes belonging to A at the output of the recognizer as received symbols, confusion matrices that characterize the data sets are then computed.

Each time a feature vector corresponding to phoneme $b_i \in B$ is passed through the trained NN, posterior probabilities corresponding to all phonemes in set A are obtained at the output of the NN. We treat each of these posterior probabilities as soft-counts to populate a phoneme confusion matrix (CM). From a fully-populated confusion matrix CM, the following counts can be derived. Entry (i, j) of the confusion matrix corresponds to the soft count aggregate $CM(i, j)$ of the total number of times phoneme b_i was recognized as a_j. Marginal count $CM(i)$ of each row is the total number of times phoneme b_i occurred in the task-specific data. Similarly count $CM(j)$ of each column is the total number of times phoneme a_j of the task-independent data set was recognized. C is the total number of counts in the confusion matrix.

Given such a CM, a useful information theoretic quantity that can be used to quantify relationships between the phonemes of each language is the *empirical pointwise mutual information* [35]. In [36], the use of this quantity in conjunction with confusion matrices has been shown. For an input alphabet A and output alphabet B, using the count based confusion matrix, the *empirical pointwise mutual information* between two symbols a_i from A and b_j from B is expressed as

$$\hat{I}_{AB}(a_i, b_j) = \log \frac{N_{ij}.N}{N_i.N_j}, \quad (1)$$

where N_{ij} is the number of times the joint event $(A = a_i, B = b_j)$ occurs and N_i, N_j are marginal counts $\sum_j N_{ij}$ and $\sum_i N_{ij}$. N is the total number of events.

Using our soft count based confusion matrix between two phoneme sets A and B, we similarly define the *empirical pointwise mutual information* between phoneme pairs (a_i, b_j) as

$$\hat{I}_{AB}(a_i, b_j) = \log \frac{CM(i, j).C}{CM(i).CM(j)}, \quad (2)$$

using quantities defined earlier.

Once an NN is trained on each language, a per-phoneme mutual information (MI) matrix for every language pair can then be computed. Entry (i, j) for one such matrix contains the MI score between phoneme i of the first language and phoneme j of the second language. We then compute the Frobenius norm of the matrix and normalize it with the total number of entries to arrive at a global MI score between the two languages.

For each of the Babel training set of languages at hand, we train a shallow 2-layer NN with 3 hours of transcribed data using context independent phonemes. After these NNs have been trained, a 10×10 MI language similarity matrix over the 10 languages is computed. The (i, j)-th entry of this matrix gives the information theoretic similarity between languages i and j

with a higher score signifying greater similarity. As shown in Fig. 3, the language similarity matrix is further used to construct a language graph from which clusters are identified as described earlier. Languages in the dominant cluster are then used to train a hierarchical multilingual feature frontend. Although languages are not jointly discriminated in this approach of constructing language similarities using individual language nets, pairwise language similarities are enhanced by converting the posterior based scored into mutual information based scores. This approach also has an advantage of being able to scale more easily as no model is jointly trained on all languages. The language similarity matrix however needs to be re-estimated (a new row/column entry needs to be estimated for each new language).

After the application of the clustering algorithm on the LID based graph, we discovered two dominant clusters - {Pashto, Tagalog, Turkish, Bengali, Assamese, Zulu} and {Lao, Haitian Creole, Tamil, Vietnamese}. The graph based on scores from individual languages on the other hand is clustered into {Pashto, Tagalog, Haitian Creole, Lao, Tamil and Zulu} and {Turkish, Bengali, Assamese, Vietnamese}. We hypothesize that the 6 language clusters discovered by the proposed techniques will be a useful representative set for extracting multilingual features. Since the technique has nearly halved the amount of training data, the multilingual frontend training time is also reduced by close to 50%. If none of the 10 languages were fully transcribed, with just 2% of the data (3 hours$\times$10), this techniques suggests that only data from 6 languages needs to be transcribed to create an effective multilingual frontend. This results in a 40% reduction in the data transcription and processing effort. In the next section, we evaluate the effectiveness of the proposed technique.

4. Experiments and Results

To evaluate the performance of multilingual frontends trained on various language groups, we use features extracted from these frontends on 6 VLLP languages - Swahili, Kurmanji, Cebuano, Kazakh, Telugu and Lithuanian, each with just 3 hours of transcribed data. Word Error Rates (WER %) are reported on a 3 hour tuning set. We start by training baseline speaker-independent (SI) acoustic models on 13-dimension PLP features with speaker-based mean and variance normalization. A context of 9 frames is spliced together and projected to a 40-dimensional feature space using linear discriminant analysis (LDA), and the class-conditional distributions are further diagonalized using a global, semi-tied covariance (STC) transform.

In the following SI multilingual step, the PLP+LDA+STC features described above are fused with ML features, transformed by LDA and STC, and then used as input for a two-fold DNN pipeline. In each fold, a new alignment is generated with the current model and a new decision tree is built on top of the alignment. The DNN training procedure comprises : (1) discriminative layer-wise pre-training and (2) training with cross-entropy criterion. The DNN comprises 3 hidden layers of 1024 ReLU units, followed by one 1024-unit sigmoid layer and a 1000-unit softmax layer. The baseline language models (LM) are Kneser-Ney (KN)-smoothed bigram models with a 5K vocabulary size. All the acoustic models are hybrid models trained using the IBM Attila speech recognition toolkit [37].

Table 1 shows the Word Error Rates (WER %) with the baseline acoustic features (PLP) and multilingual features extracted from a feature frontend trained on 10 languages (ML-10) [15]. For all the languages, multilingual features provide significant gains (up to 15% relative improvements) over the basic features. In the next set of experiments, we train a set of multilingual frontends on the clusters identified using the two techniques described earlier. These frontends have training data sampled from the full training set of 10 languages in two different ways. They include -

(a) A frontend on language clusters identified using the LID based NN (LID)

(b) A frontend on language clusters identified using mutual information scores from individually trained NNs (IL)

Table 1: WER (%) using from various multilingual frontends.

Language	PLP	ML-10	SMP	RND	LID	IL
Swahili	75.2	66.0	67.5	68.0	67.6	66.8
Kurmanji	84.1	78.2	79.5	79.5	79.1	79.2
Tok Pisin	64.8	53.8	56.2	57.1	56.7	54.8
Cebuano	78.1	70.5	72.1	72.0	71.9	71.3
Kazakh	79.1	72.9	74.0	74.5	73.7	73.5
Telugu	87.6	82.3	83.7	84.4	83.6	82.9
Lithuanian	73.0	65.9	67.4	67.4	67.2	67.2

Table 1 shows the performance of these feature frontends in comparison with conventional PLP features and multilingual features from various frontends - (i) trained on all the languages (ML-10), (ii) trained using up to 50% of data, uniformly sampled across all the 10 languages (SMP) [15] and (iii) trained on 5 randomly selected languages - Zulu, Turkish, Haitian Creole, Tagalog and Assamese (RND). The following interesting observations can be drawn from these results -

(a) With only 50% of the data, the frontends trained on the discovered clusters perform almost as well as the frontend trained on all of the data.

(b) In most cases the frontend trained using scores from individually trained NNs performs better than the frontend trained using scores from the LID based NN. This probably confirms an earlier hypothesis that the LID based system cannot perform well as the number of languages increases.

(c) Frontends trained on the identified language clusters almost always perform better than frontends trained on random selection of data.

(d) Since all these models use only 60% of the data, this result highlights the need for selecting the right set of languages for training. The training time for the proposed frontends is around 10 days compared to 21 days for the ML-10 frontends [15]. There is clearly hence a significant reduction in training time with the proposed technique as well.

5. Conclusions

In this paper we have introduced a simple technique to perform data selection across languages for building multilingual frontends using confusion matrices. With the proposed technique we identify language clusters and show that models trained on selected candidate languages can produce very comparable performances with significantly less training time and data (close to 50% reduction in both training time and data). In this work we have assumed that the frontend is built independent of the final target language. It will be useful to investigate as future work, how languages can be selected based on prior knowledge of the final target language.

6. References

[1] A. Waibel, H. Soltau, T. Schultz, T. Schaaf, and F. Metze, "Multilingual Speech Recognition," in *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, 2000.

[2] H. Lin, L. Deng, D. Yu, Y. Gong, A. Acero, and C. Lee, "A Study On Multilingual Acoustic Modeling For Large Vocabulary ASR," in *IEEE ICASSP*, 2009.

[3] S. Thomas, S. Ganapathy, and H. Hermansky, "Cross-lingual and Multi-stream Posterior Features For Low Resource LVCSR Systems," in *ISCA Interspeech*, 2010.

[4] D. Imseng, H. Bourlard, and P. Garner, "Using KL-divergence And Multilingual Information To Improve ASR For Under-resourced Languages," in *IEEE ICASSP*, 2012.

[5] Z. Tuske, R. Schluter, and H. Ney, "Multilingual Hierarchical MRASTA Features For ASR," in *ISCA Interspeech*, 2013.

[6] Huang, J. and Li, J. and Yu, D. and Deng, L. and Gong, Y., "Cross-language Knowledge Transfer Using Multilingual Deep Neural Network With Shared Hidden Layers," in *IEEE ICASSP*, 2013.

[7] G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean, "Multilingual Acoustic Models Using Distributed Deep Neural Networks," in *IEEE ICASSP*, 2013.

[8] A. Ghoshal, P. Swietojanski, and S. Renals, "Multilingual Training Of Deep Neural Networks," in *IEEE ICASSP*, 2013.

[9] S. Thomas, M. L. Seltzer, K. Church, and H. Hermansky, "Deep Neural Network Features And Semi-supervised Training For Low Resource Speech Recognition," in *IEEE ICASSP*, 2013.

[10] F. Grézl and M. Karafiát, "Combination of Multilingual And Semi-Supervised Training For Under-Resourced Languages," in *ISCA Interspeech*, 2014.

[11] N. Thang, B. Wojtek, F. Metze, and T. Schultz, "Initialization Schemes For Multilayer Perceptron Training and Their Impact On ASR Performance Using Multilingual Data," in *ISCA Interspeech*, 2012.

[12] Y. Qian and J. Liu, "Cross-Lingual and Ensemble MLPs - Strategies for Low-Resource Speech Recognition," in *ISCA Interspeech*, 2012.

[13] A. Ragni, M.J.F. Gales and K.M. Knill, "A Language Space Representation For Speech Recognition," in *IEEE ICASSP*, 2015.

[14] Y. Zhang, E. Chuangsuwanich, J. Glass, "Language ID-based Training Of Multilingual Stacked Bottleneck Features," in *ISCA Interspeech*, 2014.

[15] J. Cui, B. Kingsbury, B. Ramabhadran, A. Sethy, K. Audhkhasi, X. Cui, E. Kislal, L. Mangu, M. Nussbaum-Thom, M. Picheny, Z. Tuske, P. Golik, R. Schluter, H. Ney, M.J.F. Gales, K.M. Knill, A. Ragni, H. Wang and P. Woodland, "Multilingual Representations For Low Resource Speech Recognition And Keyword Search," in *IEEE ASRU*, 2015.

[16] M. Harper, "IARPA Babel Program," http://www.iarpa.gov/index.php/research-programs/babel, [Online; accessed 2016-03-25].

[17] "Assamese Babel Language Pack," IARPA-babel102b-v0.5a.

[18] "Bengali Babel Language Pack," IARPA-babel103b-v0.4b.

[19] "Pashto Babel Language Pack," IARPA-babel104b-v0.bY.

[20] "Turkish Babel Language Pack," IARPA-babel105b-v0.4.

[21] "Tagalog Babel Language Pack," IARPA-babel106b-v0.2g.

[22] "Vietnamese Babel Language Pack," IARPA-babel107b-v0.7.

[23] "Haitian Creole Babel Language Pack," IARPA-babel201b-v0.2b.

[24] "Swahili Babel Language Pack," IARPA-babel202b-v1.0d.

[25] "Lao Babel Language Pack," IARPA-babel203b-v3.1a.

[26] "Tamil Babel Language Pack," IARPA-babel204b-v1.1b.

[27] "Kurmanji Babel Language Pack," IARPA-babel205b-v1.0a.

[28] "Zulu Babel Language Pack," IARPA-babel206b-v0.1e.

[29] "Tok Pisin Babel Language Pack," IARPA-babel207b-v1.0e.

[30] "Cebuano Babel Language Pack," IARPA-babel301b-v2.0b.

[31] "Kazakh Babel Language Pack," IARPA-babel302b-v1.0a.

[32] "Telugu Babel Language Pack," IARPA-babel303b-v1.0a.

[33] "Lithuanian Babel Language Pack," IARPA-babel304b-v1.0b.

[34] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[35] R. Fano, *Transmission of Information: A Statistical Theory of Communication*. MIT press, 1961.

[36] C. Peláez-Moreno, A. García-Moral, and F. Valverde-Albacete, "Analyzing Phonetic Confusions Using Formal Concept Analysis," *The Journal of the Acoustical Society of America*, vol. 128, p. 1377, 2010.

[37] H. Soltau, G. Saon, and B. Kingsbury, "The IBM Attila Speech Recognition Toolkit," in *IEEE SLT*, 2010.