

# Language Recognition via Sparse Coding<sup>†</sup>

Youngjune L. Gwon<sup>1</sup>, William M. Campbell<sup>1</sup>, Douglas Sturim<sup>1</sup>, H. T. Kung<sup>2</sup>

<sup>1</sup>MIT Lincoln Laboratory

<sup>2</sup>Harvard University

gyj@ll.mit.edu, wcampbell@ll.mit.edu, sturim@ll.mit.edu, kung@harvard.edu

## Abstract

Spoken language recognition requires a series of signal processing steps and learning algorithms to model distinguishing characteristics of different languages. In this paper, we present a sparse discriminative feature learning framework for language recognition. We use sparse coding, an unsupervised method, to compute efficient representations for spectral features from a speech utterance while learning basis vectors for language models. Differentiated from existing approaches, we introduce a maximum a posteriori (MAP) adaptation scheme that further optimizes the discriminative quality of sparse-coded speech features. We empirically validate the effectiveness of our approach using the NIST LRE 2015 dataset.

**Index Terms:** speech recognition, sparse coding

## 1. Introduction

Originally used to explain neuronal activations [1], sparse coding emerges as an effective means to discover underlying structures of unknown data. High-level feature representations learned from sparse coding occasionally have resulted the best performance for discriminative tasks in computer vision. Yet, sparse coding of speech features—or audio signals in general—has not been explored extensively. In this paper, we investigate a discriminative learning framework based on sparse coding for language recognition.

Language recognition is a systematic process of identifying the spoken language in a speech utterance. Over the years, Gaussian mixture models (GMMs) [2] and support vector machine (SVM) [3] have been crucial to build a high-performance language identification (LID) system. More recently, the idea of total variability space or i-vector [4] has been studied for LID. Motivated by joint factor analysis (JFA) approach [5] for speaker verification, i-vector approaches are known to perform better than JFA.

Sparse coding has been previously applied to speaker and language identification [6, 7, 8]. Despite much interest from the machine learning community, there is surprisingly little work in sparse coding for speech. In a classification pipeline for sparse coding, a simple classifier such as linear SVM is trained on the learned sparse feature vectors and known to perform on par with (or better than) more complex nonlinear schemes (*e.g.*, deep neural networks, kernel SVM) [9]. One possible explanation is that sparse coding can achieve a near-optimal approximation of much complicated nonlinear relationship through local and piecewise linear functions.

We structure the rest of this paper as follows. In Section 2, we present a background on sparse coding. Section 3 describes our sparse coding-based approaches for language recognition. In particular, we propose adaptive sparse coding (ASC), an enhancement to the semi-supervised classification pipeline on vanilla sparse coding, and discuss an online method for per-utterance dictionary adaptation. As a result, we can significantly improve the discriminative quality of sparse-coded speech features. In Section 4, we evaluate the proposed approaches against an i-vector based benchmark pipeline developed by Lincoln Laboratory and MIT on a subset of the NIST LRE 2015 comprising the Arabic and Chinese clusters. Section 5 concludes the paper.

## 2. Sparse Coding Background

Sparse coding is an unsupervised method to learn an efficient representation of data using a small number of basis vectors. It has been used to discover higher-level features present in data from unlabeled examples. Given an example  $\mathbf{x} \in \mathbb{R}^N$ , sparse coding searches for a representation  $\mathbf{y} \in \mathbb{R}^K$  (*i.e.*, the feature vector for  $\mathbf{x}$ ) while simultaneously updating the dictionary  $\mathbf{D} \in \mathbb{R}^{N \times K}$  of  $K$  basis vectors by

$$\min_{\mathbf{D}, \mathbf{y}} \|\mathbf{x} - \mathbf{D}\mathbf{y}\|_2 + \lambda \|\mathbf{y}\|_1 \quad \text{s.t.} \quad \|\mathbf{d}_i\|_2 \leq 1, \forall i \quad (1)$$

where  $\mathbf{d}_i$  is  $i$ th dictionary atom in  $\mathbf{D}$ , and  $\lambda$  is a regularization parameter that penalizes over the  $\ell_1$ -norm, which induces a sparse solution. With  $K > N$ , sparse coding typically trains an overcomplete dictionary. This makes the sparse code  $\mathbf{y}$  higher in dimension than  $\mathbf{x}$ , but only  $S \ll N$  elements in  $\mathbf{y}$  are nonzero.

A more direct way to control sparsity is to regularize on the  $\ell_0$  pseudo-norm  $\|\mathbf{y}\|_0$ , describing the number of nonzero elements in  $\mathbf{y}$ . However, it is known to be intractable to compute the sparsest  $\ell_0$  solution in general. The approach in Eq. (1) is called least absolute shrinkage and selection operator (LASSO) [10], a convex relaxation of the  $\ell_0$  sparse coding that induces sparse  $\mathbf{y}$ 's. We use least angle regression (LARS) [11] to solve the LASSO problem. We also consider orthogonal matching pursuit (OMP) [12], a greedy- $\ell_0$  sparse coding algorithm that computes the  $\ell_0$  sparse coding problem extremely fast by

$$\min_{\mathbf{D}, \mathbf{y}} \|\mathbf{x} - \mathbf{D}\mathbf{y}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{y}\|_0 \leq S. \quad (2)$$

OMP finds at most an  $S$ -sparse  $\mathbf{y}$  explicitly.

## 3. Our Approach

### 3.1. Shifted delta cepstral feature extraction

We use a spectral-based technique by Torres *et al.* [13, 14] to process speech waveforms. Speech is analyzed with a Hamming window of 20-msec duration at a 10-msec frame rate.

<sup>†</sup>Distribution A: Public Release, unlimited distribution. This work was sponsored by the Department of Defense under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

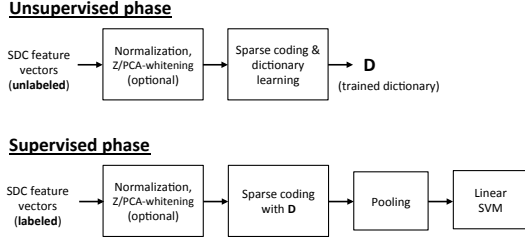


Figure 1: Vanilla sparse coding pipeline

The windowed speech waveforms pass through a mel-scale filterbank and RASTA filtering with per-utterance normalization to zero mean and unit variance. Using the 7-1-3-7 scheme, we calculate the shifted delta cepstral (SDC) coefficients. Concatenating with static cepstra, the spectral features extracted from speech form a 56-dimensional vector. Lastly, we run energy-based speech activity detection to remove undesirable background noise.

### 3.2. Vanilla sparse coding

The key reasoning for sparse coding is to learn useful representations by decomposing spectro-temporal features of speech into a *sparse* linear combination of basis vectors in a dictionary (also learned). Nonzeros in the computed sparse code quantify the presence of specific basis vectors. By exploiting variation of the nonzero locations and magnitude, we can build a discriminative pipeline for language recognition.

Figure 1 describes a baseline sparse coding approach for LID, which we call “vanilla sparse coding (VSC).” VSC is a semi-supervised approach. Assuming  $L$  languages of interest  $\mathcal{L} \in \{1, \dots, l_L\}$ , we perform sparse coding with an unbiased mix of unlabeled speech examples from all languages to train a dictionary  $\mathbf{D} \in \mathbb{R}^{N \times K}$  during the unsupervised phase. The trained dictionary represents universal sparse modeling of the  $L$  languages. That is, given an unknown speech input  $\mathbf{x} \in \mathbb{R}^N$ , we can compute its sparse representation  $\mathbf{y} \in \mathbb{R}^K$  using  $\mathbf{D}$ . By sparse modeling assumption,  $\mathbf{y}$  has only several nonzero elements

$$\mathbf{x} \approx y_1 \mathbf{d}_1 + y_2 \mathbf{d}_2 + \dots + y_K \mathbf{d}_K,$$

where  $y_j$  is  $j$ th element in  $\mathbf{y}$ ,  $\mathbf{d}_j$  the  $j$ th basis vector in  $\mathbf{D}$ . We use the notation  $\mathbf{X} = [\mathbf{x}^{(1)} \dots \mathbf{x}^{(n)}]$  for a batch of  $n$  unlabeled training examples, where  $\mathbf{x}^{(i)} \in \mathbb{R}^N$  is the  $i$ th example in the batch. Optionally,  $\mathbf{X}$  can be normalized and whitened before sparse coding for better result.

The supervised phase uses a labeled dataset. Consider  $m$  labeled training examples in  $\mathbf{X}_\ell = [\boldsymbol{\chi}^{(1)} \dots \boldsymbol{\chi}^{(m)}]$ . Now, each example  $\boldsymbol{\chi}^{(k)} = \{\mathbf{x}^{(k)}, l^{(k)}\}$  includes a language label  $l^{(k)} \in \mathcal{L}$  for  $\mathbf{x}^{(k)}$ . Recall each  $\mathbf{x}$  contains the spectral feature for a single frame (*i.e.*, 10 msec). Since a speech utterance is much longer (up to minutes), sparse coding will result in too many feature vectors per utterance. Before the supervised training of classifiers, we perform pooling, a technique popularized in computer vision, across all sparse codes from the same utterance. The purpose of pooling is two-fold: 1) aggregation of feature vectors and 2) statistical robustness.

### 3.3. Enhancement: adaptive sparse coding

We propose an enhancement of VSC as illustrated in Figure 2. We name the approach “adaptive sparse coding (ASC).” The

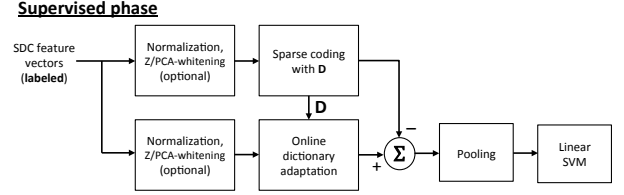


Figure 2: During the supervised phase of adaptive sparse coding pipeline, per-utterance dictionary adaptation is performed. Difference vector between adapted and universal sparse models is used to train classifiers.

unsupervised phase of ASC is identical to VSC, and the dictionary  $\mathbf{D}$  for universal sparse modeling of all languages is first learned. The basic idea of ASC is to adapt  $\mathbf{D}$  to the utterance-dependent dictionary  $\mathbf{D}_a$  during the supervised phase. With both  $\mathbf{D}$  and  $\mathbf{D}_a$ , we can compute two sparse codes  $\mathbf{y}$  and  $\mathbf{y}_a$  for each input vector  $\mathbf{x}$  from the same utterance such that  $\mathbf{x} = \mathbf{D}\mathbf{y}$  and  $\mathbf{x} = \mathbf{D}_a\mathbf{y}_a$ , respectively. ASC takes in the difference  $\Delta = \mathbf{y}_a - \mathbf{y}$  to train classifiers (compared to  $\mathbf{y}$  for VSC as in Figure 1). Note that  $\Delta$  vectors from the same utterance are also pooled before applied to classifiers.

Our idea of adapted sparse coding dictionaries and forming discriminative  $\Delta$  is analogous to adapted GMM-UBM and supervectors [2, 15]. Consider a probabilistic model for sparse coding under a Gaussian noise

$$p(\mathbf{x}|\mathbf{D}, \mathbf{y}) \sim \mathcal{N}\left(\sum_{j=1}^K y_j \mathbf{d}_j, \sigma^2 \mathbf{I}\right) \quad (3)$$

where the Gaussian noise has a zero-mean and covariance  $\sigma^2 \mathbf{I}$ . A sparse prior  $p(\mathbf{y}) \propto \prod_j e^{-\lambda|y_j|}$  regularizes the activations on sparse code  $\mathbf{y}$ . Note that the hyperparameter  $\lambda$  is same as the regularization parameter of Equation (1). We can formulate the maximum a posteriori (MAP) estimation problem to solve for  $\{\mathbf{y}_a, \mathbf{D}_a\}$  jointly

$$\arg \max_{\mathbf{D}', \mathbf{y}'} p(\mathbf{y}'|\mathbf{x}, \mathbf{D}') = \arg \max_{\mathbf{D}', \mathbf{y}'} p(\mathbf{x}|\mathbf{D}', \mathbf{y}')p(\mathbf{y}'). \quad (4)$$

Since  $p(\mathbf{x}|\mathbf{D}', \mathbf{y}')$  is a multivariate Gaussian density function, we can derive an analytical solution for Equation (4). For this paper, however, we focus on efficient estimation of the adapted dictionary and sparse code by following an online method by Mairal *et al.* [16].

In Algorithm 1, we present a fast online algorithm for dictionary adaptation. This algorithm is guaranteed to converge and computes a good estimate of  $\mathbf{D}_a$  from  $\mathbf{D}$  given an arbitrary amount of utterance input. In particular, block-coordinate descent in the inner-loop sequentially updates each basis vector (column) in the dictionary. Since the  $\mathbf{y}$  vectors are sparse, the coefficients of the matrix  $\mathbf{A}$  are concentrated on the diagonal, making the search for optimal  $\mathbf{D}_a$  very efficient. For the sparse coding step in the inner-loop, we can use either LARS or OMP.

### 3.4. SVM classification

We consider support vector machines (SVMs) for both VSC and ASC pipelines. The kernel trick for SVM has been studied widely to cope with cases where the input vectors for SVM are not linearly separable. For our case, sparse coding and pooling together give reasonably sufficient nonlinear transformation for

---

**Algorithm 1** *Online dictionary adaptation*


---

- 1: **require:** universal sparse modeling dictionary  $\mathbf{D}$  from unsupervised phase
  - 2: **initialize:**  $\mathbf{D}_a^0 := \mathbf{D}$ ,  $\mathbf{A}^0 := \mathbf{0}$ ,  $\mathbf{B}^0 := \mathbf{0}$
  - 3: **for**  $t := 1$  to  $T$  (inner-loop)
  - 4:   draw  $\mathbf{x}$  uniformly random from  $\mathbf{X}$
  - 5:   compute sparse code  $\mathbf{y}$  for  $\mathbf{x}$  using  $\mathbf{D}_a^{t-1}$
  - 6:   update  $\mathbf{A}^t := \mathbf{A}^{t-1} + \mathbf{y}\mathbf{y}^\top$  and  $\mathbf{B}^t := \mathbf{B}^{t-1} + \mathbf{x}\mathbf{y}^\top$
  - 7:   update by block-coordinate descent  
 $\mathbf{D}_a^t := \arg \min_{\mathbf{D}'} \frac{1}{t} [\frac{1}{2} \text{Tr}(\mathbf{D}'^\top \mathbf{D}' \mathbf{A}^t) - \text{Tr}(\mathbf{D}'^\top \mathbf{B}^t)]$
  - 8: **end**
  - 9: **return:**  $\mathbf{D}_a^t$
- 

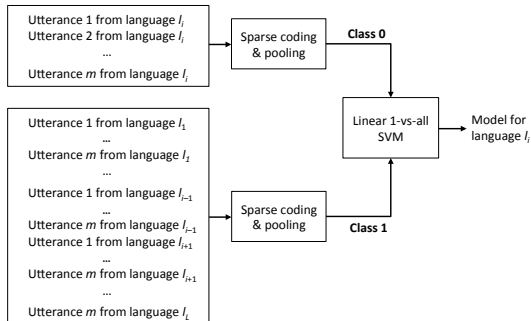


Figure 3: Training 1-vs-all SVM for each language

the SDC coefficients. Hence, we use off-the-shelf linear SVMs only.

A well-accepted strategy for a LID system is to train 1-vs-all classifiers as explained in Figure 3. To train the model for language  $l_i \in \mathcal{L}$ , we input the pooled sparse codes for all labeled examples from  $l_i$  as class 0. For utterances from all other languages  $l_j \in \mathcal{L} \setminus l_i$ , we use class 1.

## 4. Experiments

### 4.1. Task, dataset, and evaluation metrics

To evaluate the performance of sparse coding pipelines for LID, we consider the NIST Language Recognition Evaluation (LRE) 2015 [17]. The task is to determine the average performance of a LID system that can classify each language as a target within six predefined language clusters. The language clusters are Arabic, Chinese, English, French, Slavic, and Iberian with 20 different languages in total. For the time being, we present a partial evaluation focusing only on the Arabic and Chinese clusters. As summarized in Table 1, there are 5 languages from Arabic and 4 languages from Chinese in NIST LRE 2015.

The dataset comes in `train`, `test`, and `eval` subsets. We use the `train` and `test` subsets for development. The amount of development data for each language is uneven. It ranges from 2.6 (`zho-yue`) to 97.5 hours (`ara-arz`) in speech duration. The `eval` subset serves as held-out data to evaluate the classi-

Table 1: Arabic and Chinese language clusters from NIST LRE 2015

Cluster	Target languages
Arabic	Egyptian, Iraqi, Levantine, Maghrebi, Modern Standard
Chinese	Cantonese, Mandarin, Min, Wu

fication performance. Following the 2015 evaluation plan [18], we adopt the NIST average cost performance as our primary evaluation metric

$$C_{\text{avg}} = \frac{1}{N_L} \left\{ C_{\text{miss}} \cdot p_{\text{target}} \cdot \sum_{l_T} p_{\text{miss}}(l_T) + \frac{1}{N_L - 1} [C_{\text{FA}} \cdot (1 - p_{\text{target}}) \cdot \sum_{l_T} \sum_{l_N} p_{\text{FA}}(l_T, l_N)] \right\}.$$

In addition, we compute the classification accuracy metrics for the nine 1-vs-all linear SVMs on VSC and ASC.

### 4.2. Methods and training

For comparative performance evaluation, we have trained a benchmark pipeline that takes in SDC feature vectors in an  $i$ -vector framework [4], which we call “IVEC-benchmark.” IVEC-benchmark also uses a 7-1-3-7 SDC scheme along with static cepstra for the same 56-dimensional vector input to GMM-UBM and  $i$ -vector extraction. For years,  $i$ -vector based systems have been able to produce state-of-the-art results in speaker and language recognition tasks. IVEC-benchmark remains to be a part of MIT Lincoln Laboratory’s NIST LRE 2015 submission [19].

Before sparse coding, we normalize each input vector by removing its mean and dividing by the standard deviation. The normalized input vectors are then ZCA-whitened [20]. Empirically, we choose ZCA-whitening over PCA-whitening, and there is no dimensionality reduction.

We have tested multiple configurations of VSC and ASC by varying the choice of sparse coding algorithm,  $\ell_1$ -regularized LARS or greedy- $\ell_0$  OMP, and the number of basis vectors in a dictionary  $K = 512, 1024$ . For example, ASC-LARS-1024 denotes adaptive sparse coding with LARS and a 1,024-basis vector dictionary. Throughout our experiments with LARS, we use a sparsity penalty  $\lambda = 0.15$ . For OMP, we use a sparsity bound  $S = 0.1 \times 56 \approx 6$ .

During the unsupervised phase, we use the `train` subset to train  $\mathbf{D}$  for universal sparse modeling of all 9 languages. During the supervised phase, we partition `test` into five folds and do cross-validation to determine hyperparameters of the SVMs. For ASC, we also adapt  $\mathbf{D}$  to utterance-specific dictionaries with the `test` subset during the supervised phase. We have tested average and max pooling methods described below.

1. Average:  $f(\{\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(M)}\}) = \frac{1}{M} \sum_{j=1}^M |\mathbf{y}^{(j)}|$
2. Max:  $f(\{\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(M)}\}) = \max_{\forall k} (\{|y_k^{(1)}|, \dots, |y_k^{(M)}|\})$

### 4.3. Fusion

As with the historical NIST LREs, if running multiple pipelines concurrently is found beneficial, we should consider post-processing at the backend that consists of per-pipeline calibration and fusion. For example, we can do a simple linear fusion of IVEC-benchmark and one of our sparse coding pipelines:

$$llr_{\text{fusion}}^{l_T} = \rho \cdot \frac{llr_1^{l_T} - \mu_1^{l_T}}{\sigma_1^{l_T}} + (1 - \rho) \cdot \frac{llr_2^{l_T} - \mu_2^{l_T}}{\sigma_2^{l_T}}. \quad (5)$$

Here, we use a mixing ratio  $\rho$  to combine the scores (*i.e.*, log-likelihood ratios  $llr_1$  and  $llr_2$  with respect to a target language  $l_T$ ) from the two pipelines. We can also think of more sophisticated fusion schemes on logistic regression and neural networks.

Table 2: Comparison of average cost performance ( $C_{\text{avg}}$ ) on Arabic and Chinese language clusters

Pipeline	Arabic	Chinese
IVEC-benchmark	0.2566	<b>0.2054</b>
IVEC-benchmark (GMM-SAD)	<b>0.2539</b>	0.2231
VSC-LARS-512	0.2615	0.2556
VSC-OMP-512	0.2823	0.2699
VSC-LARS-1024	<b>0.2393</b>	<b>0.2043</b>
VSC-OMP-1024	0.2486	0.2120
ASC-LARS-512	0.2187	0.1909
ASC-OMP-512	0.2342	0.2036
ASC-LARS-1024	<b>0.1874</b>	<b>0.1634</b>
ASC-OMP-1024	0.2015	0.1983

#### 4.4. Results and discussion

Table 2 presents the performance comparison on the average cost metric  $C_{\text{avg}}$  for IVEC-benchmark, as well as the proposed VSC and ASC pipelines. These results are obtained by running the `eval` subset, which includes 34,530 utterances for Arabic and 44,596 utterances for Chinese. The bold-faced numbers represent the best result from each IVEC, VSC, ASC group. Notice we include the result for IVEC-benchmark under GMM-SAD, which performs better for Arabic.

We observe that ASC makes a significant improvement over VSC. If the choice of sparse coding algorithm and the number of basis vectors in a dictionary  $K$  are the same, ASC results in consistently better cost performance. Overcompleteness of sparse coding dictionary is an important hyperparameter to preconfigure. For both VSC and ASC, increasing  $K$  from 512 to 1,024 has always improved the cost performance. Also for both pipelines, LARS results in a better performance. However, the computation time for LARS is found an order of magnitude higher than OMP.

## 5. Conclusion

Sparse coding has achieved state-of-the-art performance in computer vision and object recognition. Despite its growing interest, there is relatively little work in sparse coding for acoustic language modeling. In this paper, we have described semi-supervised approaches for sparse coding on the task of language recognition. Differentiated from the existing sparse representation classification (SRC), we propose the MAP adaptation on the dictionary for sparse modeling of all languages to improve the discriminative quality of sparse-coded speech features. Using the NIST LRE 2015 dataset, we empirically evaluate the effectiveness of our approaches. Also, our experimental backend results indicate that sparse coding, ASC in particular, should be a viable component for the top LID system.

## 6. References

- [1] B. Olshausen and D. Field, "Sparse Coding with an Overcomplete Basis Set: Strategy Employed by V1?" *Vision research*, 1997.
- [2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker Verification Using Adapted Gaussian Mixture Models," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 19–41, 2000.
- [3] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-Carrasquillo, "Support Vector Machines for Speaker and Language Recognition," *Computer Speech & Language*, vol. 20, no. 2-3, pp. 210–229, 2006.
- [4] N. Dehak, P. A. Torres-Carrasquillo, D. A. Reynolds, and R. Dehak, "Language Recognition via  $i$ -Vectors and Dimensionality Reduction," in *INTERSPEECH*, 2011.
- [5] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A Study of Interspeaker Variability in Speaker Verification," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 16, no. 5, pp. 980–988, July 2008.
- [6] G. Sivaram, S. K. Nemala, M. Elhilali, T. D. Tran, and H. Hermansky, "Sparse Coding for Speech Recognition," in *ICASSP*, 2010.
- [7] T. N. Sainath, B. Ramabhadran, M. Picheny, D. Nahamoo, and D. Kanevsky, "Exemplar-based Sparse Representation Features: From TIMIT to LVCSR," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2598–2613, November 2011.
- [8] O. Vinyals and L. Deng, "Are Sparse Representations Rich Enough for Acoustic Modeling?" in *INTERSPEECH*, 2012.
- [9] A. Coates, A. Y. Ng, and H. Lee, "An Analysis of Single-layer Networks in Unsupervised Feature Learning," in *AISTATS*, 2011.
- [10] R. Tibshirani, "Regression Shrinkage and Selection via Lasso," *Journ. of Royal Statistical Society*, 1994.
- [11] B. Efron, T. Hastie, L. Johnstone, and R. Tibshirani, "Least Angle Regression," *Annals of Statistics*, 2004.
- [12] J. Tropp and A. Gilbert, "Signal Recovery From Random Measurements via Orthogonal Matching Pursuit," *IEEE Trans. on Information Theory*, 2007.
- [13] P. A. Torres-Carrasquillo, E. Singer, M. A. Kohler, R. J. Greene, D. A. Reynolds, and J. R. Deller Jr, "Approaches to Language Identification Using Gaussian Mixture Models and Shifted Delta Cepstral Features," in *INTERSPEECH*, 2002.
- [14] D. A. Reynolds, W. M. Campbell, W. Shen, and E. Singer, "Automatic Language Recognition via Spectral and Token Based Approaches," in *Springer Handbook of Speech Processing and Communication*. Springer-Verlag GmbH, 2007.
- [15] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, "SVM Based Speaker Verification Using a GMM Supervector Kernel and NAP Variability Compensation," in *ICASSP*, 2006.
- [16] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online Dictionary Learning for Sparse Coding," in *ICML*, 2009.
- [17] National Institute of Science and Technology (NIST), "2015 Language Recognition Evaluation," <http://www.nist.gov/itl/iad/mig/lre15.cfm>.
- [18] —, "The 2015 NIST Language Recognition Evaluation Plan (LRE15)," [http://www.nist.gov/itl/iad/mig/upload/LRE15\\_EvalPlan\\_v23.pdf](http://www.nist.gov/itl/iad/mig/upload/LRE15_EvalPlan_v23.pdf).
- [19] P. A. Torres-Carrasquillo, N. Dehak, E. Godoy, D. Reynolds, F. Richardson, S. Shum, E. Singer, and D. Sturim, "The MITLL NIST LRE 2015 Language Recognition System," in *Odyssey (to appear)*, 2016.
- [20] A. Coates and A. Y. Ng, "Learning Feature Representations with K-Means," in *Neural Networks: Tricks of Trade*, 2012.