2016 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 13–16, 2016, SALERNO, ITALY

INTRA-CLUSTER TRAINING STRATEGY FOR DEEP LEARNING WITH APPLICATIONS TO LANGUAGE IDENTIFICATION

Alan Joseph Bekker[1] *Irit Opher*[2] *Itsik Lapidot*[2] *Jacob Goldberger*[1]

[1]Engineering Faculty, Bar-Ilan University, Israel
[2]Afeka Center for Language Processing (ACLP), Afeka Acad. Coll. of Eng, ,Israel

ABSTRACT

In this study we address the problem of training a neural network for language identification using speech samples in the form of i-vectors. Our approach involves training a classifier and analyzing the obtained confusion matrix. We cluster the languages by simultaneously clustering the columns and the rows of the confusion matrix. The language clusters are then used to define a modified cost function for training a neural-network that focuses on distinguishing between the true language and languages within the same cluster. The results show enhanced language identification on the NIST 2015 language identification dataset.

Index Terms— Confusion matrix, clustering, language identification

1. INTRODUCTION

The purpose of automatic language recognition is to identify which language is spoken from a speech sample. There are many characteristics of speech that could be used to identify languages. Languages are made up of different sounds that form phonemes, so it is possible to distinguish languages based on the acoustic features present in the speech signal. There are of course also lexical information. Languages are separable by the vocabulary, or sets of words and syntactic rules. In this study we focus on language identification that is based solely on the acoustic information conveyed by the speech signal. The applications of language identification systems include multilingual translation systems and emergency call routing, where the response time of a fluent native operator might be critical. In the past few years, with the growing focus on deep neural networks (DNN), new training procedures [1] and optimization techniques were introduced [2],[3], showing significant improvement in computer-vision, speech recognition and language processing tasks. In particular the performance improvement obtained using DNNs for automatic speech recognition (ASR) [4] has motivated the application of DNNs to speaker and language recognition. DNNs were trained for a different purpose to learn frame-level features that are then used to train a secondary classifier for the intended language recognition task [5][6]. DNNs have also been applied to directly train language classification systems [7][8].

In this study we apply DNN to language recognition and report performance on the NIST 2015 Language Recognition i-vector Machine Learning Challenge [9]. In this task there are i-vectors examples from 50 languages. The standard DNN training procedure is based on a soft-max output layer that provides a distribution over all possible classes. The training score is the probability of the true class. In our task there are pairs of languages that are very similar to each other (e.g. Czech and Slovak) and languages that sound completely different (e.g. Spanish and Japanese). A classifier will rarely misclassify a Spanish example as Japanese but it might classify Czech as Slovak. In this study we explore a training strategy that is based on distinguishing between the true language and similarly sound languages. We first exploit the structure of the confusion matrix to cluster the language set. Both the rows and the columns of the confusion matrix correspond to the language set we want to cluster. In our approach we simultaneously cluster the row and the columns to obtain an acoustic based language clustering. Next we propose a modified cost function that encourages the neural net to learn how to distinguish between the intra-cluster examples rather than distinguishing between examples from different clusters that are assumed to be well separated. We show that the modified cost function yields improved results on the NIST-2015 challenge dataset. Furthermore we show that the proposed clustering method can be used at the training stage effectively, in terms of classification performance. Finally we analyze the differences between the linguistic and acoustic clustering approaches and show how our approach can reveal language relation information in cases where a precise linguistic description is not available.

2. INTRA-CLUSTER COST FUNCTION

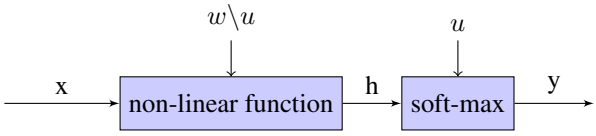
Assume we want to train a multi-class neural-network. The classifier output is a distribution on the possible classes $p(y = i|x; w)$ where $x \in \mathbb{R}^d$ is the feature vector, k is the size of the class-set, $y \in \{1, ..., k\}$ is the class label and w is the network

978-1-5090-0746-2/16/$31.00

parameter-set. The neural network classifier is based on non-linear intermediate layers followed by a soft-max output layer used for soft classification. Denote the non-linear function applied on an input x by $h = h(x)$ and denote the soft-max output layer by:

$$p(y = i|x; w) = \frac{\exp(u_i^\top h)}{\sum_{j=1}^{k} \exp(u_j^\top h)}, \quad i = 1, ..., k \qquad (1)$$

such that $u_1, ..., u_k$ are the soft-max parameters which form a subset of the entire network parameter set w. In the



training phase we are given n feature vectors $x_1, ..., x_n$ with corresponding labels $y_1, ..., y_n$. The standard cost function we aim to maximize in the training step is the log-likelihood score:

$$S_{lik}(w) = \sum_t \log p(y_t|x_t; w). \qquad (2)$$

Two classes can be well separated by the neural network or it can be difficult to distinguish between them. For instance if we are dealing with a language classification problem, a good classifier will rarely misclassify a "Spanish" example as "Japanese" but it might classify it as another Latinate language such as "Italian". Assume that we cluster the k classes into m clusters based on the confusion matrix (in the next section we suggest a specific algorithm). Denote the cluster index of a class i by $c(i)$. We can use the clustering to define a modified score for training the NN. This score is aimed at exploiting the structure of the confusion patterns by encouraging the neural net to learn how to distinguish between the intra-cluster examples rather than distinguishing between examples from different clusters that are assumed to be well separated. Given the network parametric structure (1) we can compute the posterior probability of the correct label given the label cluster:

$$p(y = i|x, y \in c(i); w) = \frac{\exp(u_i^\top h(x))}{\sum_{j \in c(i)} \exp(u_j^\top h(x))}. \qquad (3)$$

Using Eq. (3) we define the following intra-cluster score:

$$S_{ic}(w) = \sum_t \log p(y_t|x_t, c(y_t); w). \qquad (4)$$

In this cost function we do not waste the information conveyed in the labeled data to train the classifier to distinguish between unrelated classes. Instead we focus on the similar classes which are often to be confused.

Dropout is an algorithm for training neural networks by randomly dropping hidden units during training to prevent

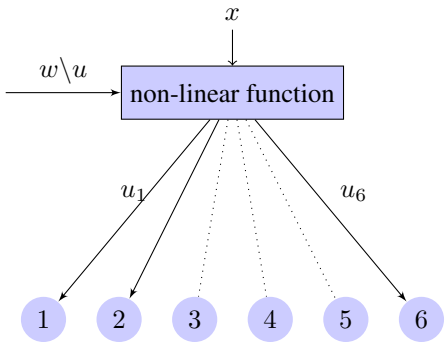


Fig. 1. The dropout soft-max layer, where the edges that do not belong to the cluster of the trained label are dropped. In this illustration the correct class is 1 and $c(1) = \{1, 2, 6\}$.

their co-adaptation [1]. The cost function $S_{ic}(w)$ (4) can be viewed as a dropout procedure applied to the units of the soft-max output layer. In each training example the network has a different topology of the soft-max layer in such a way that only classes within the cluster of the true label are allowed. The output layer dropout is illustrated in Figure 1. The dropout procedure eliminates neurons randomly. In our case, however, we eliminate outputs in a deterministic way. We only consider classed that are within the cluster of the true class.

The proposed intra-cluster cost function $S_{ic}(w)$ focuses on the more frequent intra-cluster errors. A possible drawback of the proposed cost function is that it does not take out-of-cluster errors into consideration. To deal with both error types, the score function we actually optimize is a combination of the log-likelihood score (2) and the intra-cluster score (4), i.e.:

$$S(w) = \alpha S_{lik}(w) + (1-\alpha) S_{ic}(w) \qquad (5)$$

where $0 < \alpha < 1$ balances the two components of the score function. The parameter α can be tuned by a cross-validation procedure. The partial derivative of the function $S(w)$ (5) with respect to the soft-max parameters is:

$$\frac{\partial S}{\partial u_i} = \sum_{t=1}^{n} (1_{\{y_t = i\}} - p_{ti}) h(x_t) \qquad (6)$$

s.t.

$$p_{ti} = \alpha p(y_t = i|x_t; w) + (1-\alpha) p(y_t = i|x_t, c(y_t); w).$$

At test time we do not know the label or its cluster and we use the soft-max layer over all possible labels.

The proposed method resembles the hierarchical classification approach where in the first step the classifier decides on the label group and then finds a specific label within a group. Hierarchical models for classification, however, suffer from the fact that they have to make intermediate decisions prior

to reach a final category. These intermediate decisions lead to the error propagation phenomenon that cause a decline in accuracy. Here we do not use the class categories for hierarchical decisions. Instead, we still apply a flat decision procedure that addresses intra-cluster and out-of-cluster errors in different ways.

3. CLUSTERING THE CONFUSION MATRIX

In this section we present a clustering method that is suitable for the intra-cluster training method described above. Assume we are given a classification task with features $x_1, ..., x_n \in \mathbb{R}^d$ and corresponding labels $y_1, ..., y_n \in \{1, ..., k\}$. Our goal is to find a clustering algorithm of the class-set $C = \{1, ..., k\}$ based on the labeled data. One possible approach is to use the feature vectors and their labels and apply a semi-supervised procedure of clustering the feature vectors given the constraint that vectors with the same label should be assigned to the same cluster [10]. This is an indirect approach since the clustering is applied to the feature vectors and not to the category set C. Another approach is to build a parametric model for all the features of a given class and then cluster the models. Yin et al. [11] represents the features of each class by a Gaussian mixture model and then applied a clustering procedure on the GMMs.

Our clustering is based not just in the labeled training data but also on the classifier we want to train. We first use the labeled feature vectors to train a classifier. We then compute the confusion matrix that summarizes the classifier's performance. The confusion matrix for a classifier is a square $k \times k$ matrix M such that $M(i, j)$ denotes the number of examples of class i that were classified as class j. The rows of the confusion matrix correspond to the categories we want to cluster. The gist of our approach is to directly cluster the class-set via clustering the rows of the confusion matrix.

One intuitive approach to using the confusion-matrix is to view each row as a feature representation of its corresponding class. We can then apply any clustering method to the matrix rows. For example we can apply a k-means algorithm based on the Euclidean distance between the rows. Since each row defines a distribution on the categories, we can also apply information theory based clustering algorithms that use the KL-divergence between the rows [12][13]. One problem with this strategy is that the diagonal of the confusion matrix is very dominant and therefore the rows are not expected to be similar. Another special characteristic of our setup is that both the rows and the columns of the confusion matrix correspond to the category set we want to cluster. Hence, any clustering of the matrix rows induces a clustering of the matrix columns. The category clustering approach we describe below is based on simultaneous clustering of the rows and the columns of the confusion matrix.

We start with a pre-processing step applied to the confusion matrix that eliminates the contribution of the matrix diagonal. We define a conditional distribution over a pair of categories as follows: for each two different categories a and b define:

$$P_{ab} = \frac{M(a,b)}{\sum_{c \neq a} M(a,c)}, \qquad a, b \in C \tag{7}$$

and for each category $a \in C$ define $P_{aa} = 0$. P_{ab} is the conditional probability, given that the true label is a, and that the classifier made a wrong assignment, that the classifier decision is b. Let X_1 and X_2 be random variables (r.v.) defined on the category set such that X_1 is associates with the correct label and X_2 described the erroneous label decision by the classifier. The modified confusion matrix P, by definition, is the conditional distribution of X_2 given X_1, i.e.,

$$p(X_2 = b | X_1 = a) = P_{ab}, \qquad a, b \in C. \tag{8}$$

Assuming a prior uniform distribution over the categories, i.e.

$$p(X_1 = a) = \frac{1}{k}, \qquad a \in A$$

we obtain the joint distribution of X_1 and X_2.

Let $\{A_1, ..., A_m\}$ be a partition of the k object categories into m clusters. The probability that both the random variable X_1 and X_2 will be in cluster A_i is:

$$p(X_1 \in A_i \text{ and } X_2 \in A_i) = \tag{9}$$

$$\sum_{a \in A_i} p(X_2 \in A_i | X_1 = a) p(X_1 = a) = \frac{1}{k} \sum_{a \in A_i} \sum_{b \in A_i} P_{ab}.$$

The intuitive clustering score, which we want to maximize, is the probability that the true category and the classified category are in the same cluster, i.e.:

$$\text{Score}(A_1, ..., A_m) = \sum_{i=1}^{m} p(X_1 \in A_i \text{ and } X_2 \in A_i). \tag{10}$$

However, the clustering that maximizes this criterion is the one formed by a single cluster that contains all the data points. Even if we require all the m clusters to be non-empty, this score still favors partitions that are very unbalanced. To overcome this tendency, we define the following score function:

$$\begin{aligned} \text{Score}(A_1, ..., A_m) &= \sum_{i=1}^{m} p(X_2 \in A_i | X_1 \in A_i) \\ &= \sum_{i=1}^{m} \frac{\frac{1}{k} \sum_{a \in A_i} \sum_{b \in A_i} P_{ab}}{\frac{|A_i|}{k}} = \sum_{i=1}^{m} \frac{1}{|A_i|} \sum_{a \in A_i} \sum_{b \in A_i} P_{ab} \end{aligned} \tag{11}$$

This score measures the probability that frequently confused labels are clustered together. The intuition behind this score is that in a good clustering we expect that even when a classifier makes a wrong category decision, the classification result is still expected to be in the same cluster of the correct object

category. In other words, we look for a category partitioning such that an object classifier seldom transitions from one category cluster to another. The optimal clustering is the one that maximizes the clustering score (11). Note that the score (11) does not encourage all the clusters to have the same size. It is also not necessarily optimized by having a huge cluster that includes almost all the categories.

Maximizing the score (11) is known to be NP hard even for $m = 2$ [14]. Variants of the spectral clustering algorithm can be applied to find an optimal solution for a relaxation of the proposed criterion (11). Here we used a greedy sequential optimization approach that was found to work better than spectral methods in terms of both performance and computational complexity [15]. The sequential clustering algorithm starts with a random clustering of the k categories into m clusters. We then go over the data points in a circular manner and check for each point whether its removal from one cluster to another can increase the clustering score (11). This loop is iterated until no single-point transition offers an improvement. Since there is no guarantee that the algorithm will find the global optimum, we can run the algorithm using several random initial partitions and choose the best local optimum. Alternatively we can use a multi-level clustering or a bottom-up agglomerative approach.

In the case where the category set is large we can compute the basic optimization step in an efficient way. In the basic step of the greedy optimization algorithm we remove a category a from its current cluster and assign it to a cluster such that the cost function (11) of the obtained clustering is maximized. Since only the clustering affiliation of one category is changed, most clusters remain unchanged and we do not need to recompute the entire clustering score. We dub the proposed object category algorithm which as the Confusion matrix Clustering algorithm (CMC).

Note that the score definition (11) is related to the classical normalized-cut (Ncut) criterion for pairwise clustering [16] [17]. Suppose we are given m objects and a symmetric notion of pairwise similarity $w_{ij} \geq 0$. The goal of pairwise clustering is to divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other. The main differences between pairwise data clustering and our category clustering problem is that in the pairwise clustering setup the input is provided by a symmetric similarity score between object pairs. The score is then transformed into a joint object distribution by the Laplacian representation of the data graph. In our problem the (non-symmetric) relations between categories are provided by the confusion matrix which is a conditional distribution defined on pairs of categories.

Input: Training data $x_1, ..., x_n \in R^d$ with corresponding labels $y_1, ..., y_n \in \{1, ..., k\}$.

Algorithm:

1. Train a DNN and compute the $k \times k$ confusion matrix:

$$M_{ab} = \frac{\sum_t p(y = b|x_t; w)}{\sum_t 1_{\{y_t = a\}}}, \qquad a, b = 1, ..., k$$

2. Eliminate the diagonal of the confusion matrix:

$$P_{ab} = \frac{M_{ab}}{\sum_{c \neq a} M_{ac}}, \qquad a, b = 1, ..., k$$

3. Choose a random partition $A_1, ..., A_m$ of the set $\{1, ..., k\}$.

4. Loop over the k classes until there is no change
   - Move class a into the cluster such that the clustering score (11) is maximized.

5. Train a DNN using the cost function (5).

Table 1. Training a DNN based on the Confusion Matrix Clustering (CMC) algorithm.

4. EXPERIMENTS

The NIST 2015 language recognition challenge [9] covers 50 target languages labeled data (300 speech segments per language). The speech duration of the audio segments used to create the i-vectors for the challenge were sampled from a log-normal distribution with a mean of approximately 35s. The speech segments were derived from conversational telephone and narrow-band broadcast speech data. Each speech segment is represented by an i-vector of 400 components [18]. The NIST challenge also contains an unlabeled dataset that was not used in this study. The classifier we used here is a two fully connected hidden layer Deep Neural-Network (DNN) comprising 200 and 100 neurons each and a soft-max output layer. The activation function was set to be ReLU and the optimization procedure used was mini-batch stochastic gradient descent with momentum. We also used a Dropout with a ratio of 0.5 to prevent over-fitting [1]. The fifty-class labeled dataset was used for evaluation with a 10-fold cross-validation. In the experiments described below we first applied the proposed clustering on the DNN confusion matrix results. Then we used this clustering to form a modified cost function for neural-network training.

4.1. Language Clustering

In order to analyze the data linguistically, we used the Ethnologue Catalog of Languages [19], that provides statistics and information on more than 7000 living languages. The NIST

Table 2. Linguistic Families.

Cluster	Languages	(Sub)Family
1	Hausa, Somali, Oromo, Arabic, Amharic	Afro-Asiatic
2	Indonesian, Tagalog	Austronesian
3	Ukrainian, Polish, Slovak, Czech, Russian, Bosnian	Balto-Slavic
4	Hindi, Urdu, Punjabi, Bengali	Indo-Aryan
5	Pashto, Kurdish, Tajik, Farsi, Dari	Iranian
6	Romanian, French, Portuguese, Spanish	Italic
7	Shona, Swahili, Zulu	Niger-Congo
8	Burmese, Cantonese, Mandarin, Tibetian	Sino-Tibetan
9	Laotian, Thai	Tai-Kadai
10	Tatar, Turkish, Kyrgyz, Uzbek, Azerbaijani, Kazakh	Turkic
11	Georgian, Greek, Japanese, Khmer, Kosovo, Creole , Armenian, Korean	Singleton

Table 3. Language clustering based on the CMC algorithm

Cluster	Languages
1	Kurdish, Oromo,Amharic,Somali, Arabic, Hausa
2	Punjabi, Bengali, Pashto, Tagalog, Zulu, English
3	Ukrainian, Polish, Slovak, Czech, Russian
4	Hindi, Urdu
5	Tajik, Farsi, Dari
6	Creole, French
7	Shona, Swahili
8	Tibetan, Cantonese, Mandarin
9	Burmese, Khmer, Laotian, Thai
10	Tatar, Turkish, Kyrgyz, Uzbek, Azerbaijani, Kazakh
11	Georgian, Greek, Japanese, Portuguese, Romanian, Kosovo, Indonesian, Bosnian, Spanish, Armenian, Korean

dataset contains 50 languages which, from a purely linguistic point of view, can be grouped into 11 clusters, each representing a different subfamily, such as Balto-Slavic or Afro-Asiatic. There were also 8 singleton languages of two kinds: languages that constitute a family of their own (no known sister languages), such as Japanese, and languages that belong to a broader linguistic family, such as English which is a Germanic language, but were the only representative of this family in our data set. Another interesting unique member of the training set is Creole, which is basically a family of languages. Table 2 shows a possible knowledge based grouping of the training set languages according to linguistic knowledge. Note that some of these clusters can be further split into finer clusters, as is the case of the Balto-Slavic cluster in our table, which is composed of 3 sub-clusters: Slavic East (Ukrainian & Russian), Slavic South Western (Bosnian) and Slavic West (Polish, Czech & Slovak). The latter can be split again into Lechitic (Polish) and Czech-Slovak (Czech & Slovak).

Since the cost function we defined in Eq. (11) is not convex, the optimization procedure as defined in the previous section does not guarantee achieving the global maximum. In the general case we can try several standard unsupervised initializations. In our case we already have a linguistic based clustering that can serve as a good initialization point. This way we can also incorporate linguistic features into our CMC clustering via initialization, as was done in [20]. The results of this procedure are shown in Table 3.

As mentioned above, Creole is a not a single language, but rather a general term for about 100 such languages, representing the evolution of a pidgin language which is a mixture of languages. In our dataset, there was no information regarding the nature of the Creole in the dataset, hence Creole was assigned to a singleton cluster, as it was impossible to classify it linguistically from the NIST Challenge description. Looking at the acoustic clustering revealed that the Creole used in our dataset is a French family based Creole.

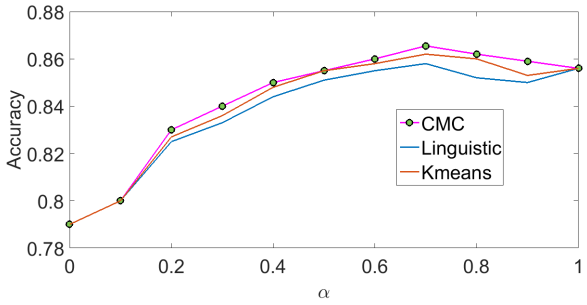


Fig. 2. Language classification accuracy as function of the α parameter for several language grouping methods.

4.2. Language Classification

We next illustrate the usage of language clustering to improve the performance of a language identification neural-network. We trained a language identification system using the intra-cluster cost function (5). Figure 2 depicts the performance of the proposed intra-cluster cost function (5) as a function of α using cross-validation. We also show classification results based on two other language groupings. The first grouping is based on linguistic similarity, as described in Table 2. The second grouping was created by applying a k-means algorithm on the 50 vectors carted by averaging all the i-vectors of each language. As can be seen in the plot, the best classification result was achieved around $\alpha = 0.7$ for all the methods and this value was used for the analysis described below. The value $\alpha = 1$ corresponds to the standard training method based solely on likelihood optimization (2). Figure 2 demonstrates that training a classifier based on the proposed cost-function (5) outperforms standard likelihood-based training. Further, the clustering based on the CMC method yields better results than the alternatives. We also tried a random grouping which achieved no improvement at all.

Given a language clustering, two types of language classification errors can be defined. The classification error result can be in the same cluster of the correct language (intra-cluster error) or it can belong to another cluster (inter-cluster error). Table 4 presents the inter-cluster and intra-cluster error analysis for the three language clustering methods described above. It also shows how the errors in the standard training method (without any clustering) are split between intra-

Table 4. Language classification error results for several language-grouping methods.

Clustering Method	Classification Error		
	intra	inter	total
no clustering (CMC,$\alpha = 1$)	10.3	4.1	14.4
linguistic	7.4	6.8	14.2
k-means	7.3	6.5	13.8
CMC	6.8	6.7	**13.5**

and inter-cluster errors according to the clusters defined by the CMC method. As could be expected, most of the errors of the standard training method are within the same cluster of the correct language. The clusters of the CMC algorithm were built exactly to minimize the inter-cluster error as detected in the confusion matrix of the baseline method. In all other methods there is the same proportion of intra-cluster and inter-cluster errors. Table 4 also shows that, for all the language-clustering methods we compared, training neural-network based on the CMC algorithm achieved the best results.

5. CONCLUSION

To conclude, this study introduced a modified cost function for training a neural-network that improved language classification results. We also proposed a language clustering method based on the confusion matrix of a classifier. We focused here on the tasks of language clustering and language identification. However, the methods proposed here are general and can be applied any to classification task.

6. REFERENCES

[1] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[2] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th international conference on machine learning (ICML-13)*, 2013, pp. 1139–1147.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Int. Conference on Machine Learning (ICML)*, 2015.

[4] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 8, pp. 82–97, 2012.

[5] Y. Song, B. Jiang, Y. Bao, S. Wei, and L.-R. Dai, "I-vector representation based on bottleneck features for language identification," *Electron. Lett.*, pp. 1569–1580, 2013.

[6] P. Matejka, L. Zhang, T. Ng, H. S. Mallidi, O. Glembek, J. Ma, and B. Zhang, "Neural network bottleneck features for language identification," in *IEEE Odyssey*, 2014, pp. 299–304.

[7] I. Lopez-Moreno, J. Gonzalez-Dominguez, D. Martinez O. Plchot, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," in *ICASSP*, 2014, pp. 5374–5378.

[8] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 2015.

[9] "NIST language recognition i-vector machine learning challenge," `https://ivectorchallenge.nist.gov/`, 2015.

[10] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, "Constrained K-means clustering with background knowledge," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2001, pp. 577–584.

[11] B. Yin, E. Ambikairajah, and F. Chen, "Improvements on hierarchical language identification based on automatic language clustering," in *ICASSP*, 2008, pp. 4241–4244.

[12] N. Tishby, F. Pereira, and W. Bialek, "The information bottleneck method," in *Allerton Conf. on Communication, Control and Computing*, 1999.

[13] N. Slonim, N. Friedman, and N. Tishby, "Unsupervised document classification using sequential information maximization," in *Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2002.

[14] D. Wagner and F. Wagner, "Between min cut and graph bisection," in *Proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science, Springer*, 1993, pp. 744–750.

[15] I. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors: A multilevel approach," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 11, pp. 1944–1957, 2007.

[16] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 22 (8), pp. 888–905, 2000.

[17] M. Meila and J. Shi, "A random walks view of spectral segmentation," in *AISTATS*, 2001.

[18] N. Dehak, P. A. Torres-Carrasquillo, D. Reynold, and R. Dehak, "Language recognition via Ivectors and dimensionality reduction," in *Interspeech*, 2011.

[19] "Ethnologue," `https://www.ethnologue.com/`.

[20] R. Caruana, M. Elhawary, N. Nguyen, and C. Smith, "Meta clustering," in *International Conference on Data Mining,*. IEEE, 2006, pp. 107–118.