Phonotactic Language Recognition using i-vectors and Phoneme Posteriogram Counts

Luis Fernando D'Haro[1], *Ondřej Glembek*[2], *Oldřich Plchot*[2], *Pavel Matejka*[2,] *Mehdi Soufifar*[2], *Ricardo Cordoba*[1], *Jan Černocký*[2]

[1]Speech Technology Group. Dept. of Electronic Engineering. E.T.S.I. Telecomunicación. Universidad Politécnica de Madrid. Ciudad Universitaria s/n, 28040- Madrid, Spain
[2]Brno University of Technology, Speech@FIT group and IT4I Center of Excellence, Czech Republic
{lfdharo,cordoba}@die.upm.es, {glembek,iplchot,matejkap,cernocky}@fit.vutbr.cz, qsoufifar@stud.fit.vutbr.cz

Abstract

This paper describes a novel approach to phonotactic LID, where instead of using soft-counts based on phoneme lattices, we use posteriogram to obtain *n*-gram counts. The high-dimensional vectors of counts are reduced to low-dimensional units for which we adapted the commonly used term i-vectors. The reduction is based on multinomial subspace modeling and is designed to work in the total-variability space. The proposed technique was tested on the NIST 2009 LRE set with better results to a system based on using soft-counts (Cavg on 30s: 3.15% vs 3.43%), and with very good results when fused with an acoustic i-vector LID system (Cavg on 30s acoustic 2.4% vs 1.25%). The proposed technique is also compared with another low dimensional projection system based on PCA. In comparison with the original soft-counts, the proposed technique provides better results, reduces the problems due to sparse counts, and avoids the process of using pruning techniques when creating the lattices.

Index Terms: subspace modeling, multinomial distributions, LID

1. Introduction

Nowadays there are two main approaches to the spoken language recognition task (LRE): a) based on acoustic, and b) on phonetic information. In the acoustic systems, different features – such as short-term spectra, prosodic information or intonation – are taken into account to model each recognized language. On the other hand, phonotactic systems model sequences of recognized phonemes obtained from a phone recognizer [1]. The focus of this paper is on the latter approach, although experiments considering the fusion and calibration with an acoustic-based system have also been done for comparison.

The front-end of a phonotacic system consists of a phone recognizer that tokenizes speech utterances into discrete events (phones), which are used to extract n-gram counts statistics. These n-gram counts can be used as feature vector for training a language model for each language (as in the case of PRLM [1]) or using a discriminative classifier (as in the case of support vector machines (SVM) [3]). In the case of PRLM, typically the language models are created using the n-gram counts applying smoothing techniques as the ones used for training a speech recognition system [4]; an alternative approach is to use soft counts (i.e. posterior-weighted counts) created from phone lattices [5] instead of 1-best phone strings and then to train a generative model as classifier. On the other hand, in case of using discriminative classifiers, it is necessary to represent the input as a fixed-length vector whose size depends on the number of phones used by the phoneme recognizer and considering the full expansion of all possible combinations of them up to a given *n*-gram order. The size of the vector limits the order of the *n*-grams considered (typically only up to trigrams or a reduced set in case of four-grams); making it necessary to look for compact representations (i.e. a dimensionality reduction) such as principal component analysis (PCA) or selection of discriminative *n*-grams as proposed in [6] and [7]. In this paper, we follow the same approach as in [9] and [10], where i-vectors are used for obtaining a compact representation of *n*-gram statistics.

Although i-vectors were first introduced for the acoustic speaker recognition task [8] with continuous features and Gaussian mixture modeling (GMM), they have being successfully used on the LID task and also extended to be trained on discrete features [10] by using subspace multinomial models to model the discrete representation of prosodic features. In our case, since we are dealing with another discrete representation of speech utterances: posteriogram-based counts from the output of the phone recognizer, and considering the results reported in [9], we decided to use the same approach.

Our new system is inspired in two different phonotactic systems reported in [12] and [9]. In [12] a phonotactic-based LID system is created using *n*-gram soft-counts created from phone lattices. These counts are used to create a vector where all possible *n*-grams combinations are present and where those *n*-grams that do not occur are set to zero. Then, a PCA projection is done in order to reduce the size of the input vector taking the most relevant information from these counts. Then, a classification system is created by using Logistic Regression (LR). In [9], a similar procedure is followed but the classification is done by first extracting i-vectors using subspace multinomial distributions (SMD) and then using them as feature vectors for training a logistic regression classifier.

In this paper, we modify mainly the input feature vectors and backend by using a multiclass logistic regression classifier. We analyze the performance of the proposed features on the i-vector paradigm with respect to the previous systems: i.e. w.r.t the performance on creating i-vectors from the soft-counts and using a PCA-based dimensionality reduction. The experiments are carried out on the NIST LRE 2009 task and all results are given in terms of the average decision cost function (C_{avg}) according to the NIST LRE 2009 evaluation plan.

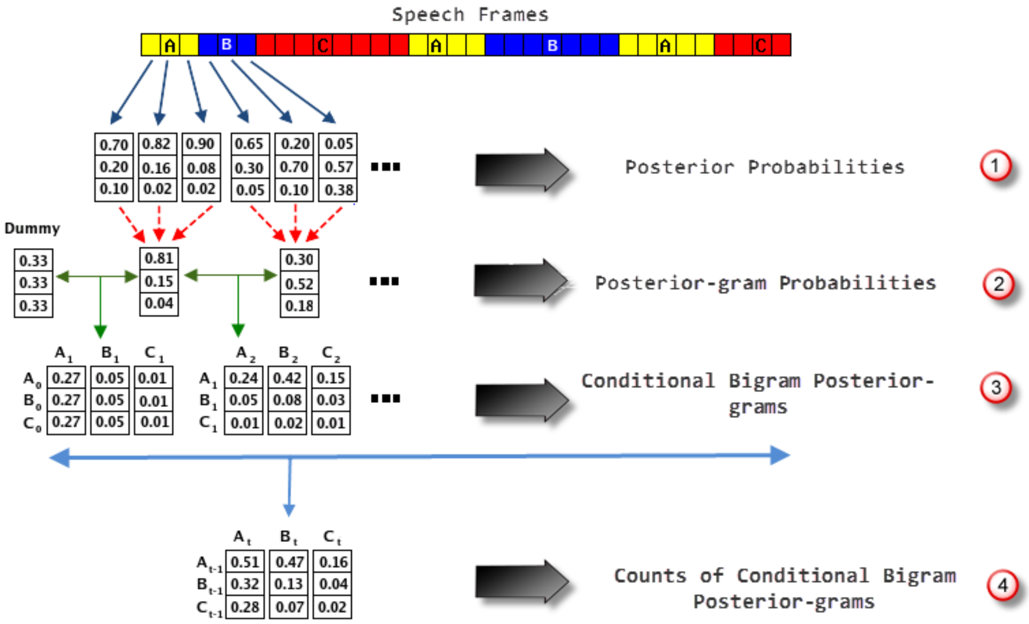

Figure 1. Example for the process of creating posteriogram-based bigram counts.

2. Joint-posteriogram *n*-gram Counts

Figure 1 shows the process of creating the vector of posteriogram-based *n*-gram counts. In this case, we consider bigram counts for simplicity. The process can be divided into four main steps.

The first step is to tokenize speech by the means of running a phone-recognizer that, for each frame, provides the posterior probabilities of the phone occurrences (number 1 in Figure 1). In our experiments, we used the BUT Hungarian phone recognizer[1].

The second step is to sum up and average the posterior probabilities for the frames that are considered to be within the same phoneme unit (number 2 in the figure). The phone boundaries are obtained by running a viterbi decoding on the posteriogram (HTK was used here). This fact can be considered as the incorporation of a priori information (i.e. the best result for obtaining phone-boundaries) to the system that we believe helps to improve the results. The reason for doing this is that the posterior probabilities for each frame are highly correlated within a single phoneme. Also we believe that the averaged posterior is robustly estimated. In the figure, we have represented the phone boundaries with different colors. The resulting entity is referred to as an *averaged posteriogram*.

The next step is to create the *joint-posteriogram* – a sequence of matrices of joint probabilities for the *n* consecutive frames. In order to do this we take the averaged posteriogram of each frame and we do outer product with the posteriogram of the previous frame. If we assume that the frames of the averaged posteriogram are statistically independent, then we are computing the joint probabilities for sequences of phonemes $p(a_{i-1},b_i)$=p(a_{i-1})p(b_i). For calculating the joint-posteriogram for the first phone we create a dummy probability vector with equal probabilities among all the phones (in the example, since we have only three phoneme,s the probability for each one is set to 0.33). This process is repeated for all the phone-grams considering the *n*-1 phone-gram history.

The final step (number 4 in the figure) is to sum up all frames (matrices) of the joint-posteriogram. This way, we create a matrix of *n*-gram counts that is converted into a 1xD vector (where D is the total number of possible *n*-grams) and then used as a feature file for training the i-vectors as explained below.

3. Subspace Multinomial Model

The goal of the Subspace Multinomial Model is to model the discrete representation of the posteriogram counts created in the previous step in a similar way as is done in [9] for *n*-gram counts in language recognition or prosodic features [10] in a speaker recognition task. Thanks to the Subspace Multinomial Models we can train low dimensional vectors of coordinates in total variability subspace, i.e. i-vectors, and then use these i-vectors as feature input for training a discriminative LID classifier. This section describes the motivation and process for training the subspace multinomial models with details in [10] and [13].

3.1. Likelihood function

The log-likelihood of data *D* for a multinomial model with *C* discrete classes is determined by model parameters φ and sufficient statistics γ, representing the occupation counts of classes for all N utterances in *D*:

$$\log p(D) = \sum_{n=1}^{N}\sum_{c=1}^{C}\gamma_{nc}\log\varphi_{nc} \qquad (1)$$

[1] http://speech.fit.vutbr.cz/software/phoneme-recognizer-based-long-temporal-context

Where γ_{nc} is the occupation count for class c and utterance n and φ_{nc} are probabilities of (utterance dependent) multinomial distribution, defined by a subspace model according:

$$\varphi_{nc} = \frac{\exp(m_c + t_c w_n)}{\sum_i^C \exp(m_i + t_i w_n)} \quad (2)$$

Where t_c is the c-th row of subspace matrix T and w_n is an r dimensional column vector (i-vector) representing language and channel of utterance *n*.

3.2. Training procedure

For training the i-vectors we have followed the algorithm reported in [10] with slight modifications in order to iterate several times the estimation and maximization step. Newton-Raphson algorithm is used for optimization. The training process is divided into three steps:

- Train the i-vector extractor matrix (T) following the algorithm in List 1. In our case, the number of epochs (E) and iterations (I, J) were set as E=4, I=J=3. In contrast to [10], we retrain matrices T and w by using the updates from the previous epoch. This way, we obtain a better convergence to the optimal value. In addition, in case of failing to improve the likelihood when updating both w and T due to making too large the update step, we halve the update step until an increase in likelihood is obtained again.

```
initialize $T_0 = random$
for $e$ in epoch 1 to E
  initialize $w_e$ =0
  for $i$ in iteration 1 to $I$
    re-estimate $w_e^i$ using $T_{e-1}^J$
  for j in iteration 1 to J
    re-estimate $T_e^j$using $w_e^I$
```

List 1. Algorithm for training i-vectors and extractor matrix

- Extract an i-vector (i.e. w) for each file of the train, development and test sets using the previously trained T matrix and re-estimating it three times.
- Finally, before applying the calibration and fusion (see section 4.3) we apply mean removal and length normalization (i.e. subtracting the mean of each i-vector and dividing it by its norm) in order to model better the low dimensional distribution where the i-vectors are located.

As explained in [10], it is possible to consider a subspace model using a single multinomial distribution (Equation 2) or using a set of multinomial models. In our experiments, we have considered a set of 1089 multinomial models when using trigrams (i.e. considering all the possible number of bigram histories, 33x33, using 33 phones for the Hungarian recognizer). We achieve this by concatenating the distributions into single super-vector of multinomial distributions, which is modeled by one subspace matrix T. In other words, there will be only one i-vector w_n defining the whole set of multinomial distributions for each segment n. In this case, the indices c from Equation (2) are divided into subsets, where each subset corresponds to mutually exclusive events (counts from one GMM), and the denominator is also changed to normalize only over the appropriate subset of indices that the current c belongs to.

4. Experimental Setup

4.1. Training and Development Data

The training data has being taken from the same databases as in [11], i.e. using speech files from Callfriend, Fisher English Part 1 and 2, Fisher Levantine Arabic, HKUST Mandarin, Mixer (data from NIST SRE 2004, 2005, 2006, 2008). For training, we used a balanced dataset with a maximum of 500 utterances per language considering the 23 target languages as defined for NIST LRE 2009[1]. This set was used for training the i-vector extractor and also for training the classifier.

The calibration back-end described in section 4.3 was trained using a development dataset, which comprises data from NIST LRE 2007, Foreign Accented English, OGI-multilingual, SpeechDat-East, OGI 22 languages, Voice of America radio broadcast and Switch Board. Again, only data of the 23 target languages were used. This set was based on segments of previous NIST LRE evaluations plus additional segments extracted from CTS, VOA3 and human-audited VOA2 data, not contained in the training dataset, see [14] for details.

4.2. Feature Extraction for Acoustic System

Standard 7 Mel Frequency Cepstral Coefficients (MFCC) (including C0) are used. Vocal Tract Length Normalization (VTLN) and Cepstral Mean and Variance Normalization is applied in MFCC computation. Then, Shifted Delta Cepstra (SDC) coefficients with usual 7-1-3-7 configuration are obtained, and concatenated to MFCCs, to obtain a final feature vector of 56 coefficients. For each utterance, the corresponding feature sequence is finally converted to an i-vector using an T matrix based on a GMM with 2048-components trained on pooled features from all 54 languages included in our training data.

4.3. Classifier and Calibration Back-end

As classifier for our i-vectors, we have used a Multiclass logistic regression which generates 23 different classifiers, one for each language. Then, these classifiers are used to generate scores for the files in our test set. For calibration and fusion, a Gaussian Back-end followed by a Discriminative Multi-Class Logistic Regression is used to post-process the scores obtained before.

5. Results

Figure 2 shows the results obtained with our proposed system (first row) for three given conditions: 30, 10, and 3 seconds and different i-vector dimensions. If we compare them with the results (second row in Table 1) obtained using also an i-vector-based phonotactic system but trained on soft-counts (using a similar setup as the one reported in [9] but with some differences on the training list and fusion system), we can see that our results are better for all the conditions. Finally, we can also try to compare this results with the ones obtained using a PCA reduction to dimension 1000 (third row in Table 1), where we can see that our system provides better results in all cases.

[1] http://www.itl.nist.gov/iad/mig/tests/lre/2009/LRE09_EvalPlan_v6.pdf

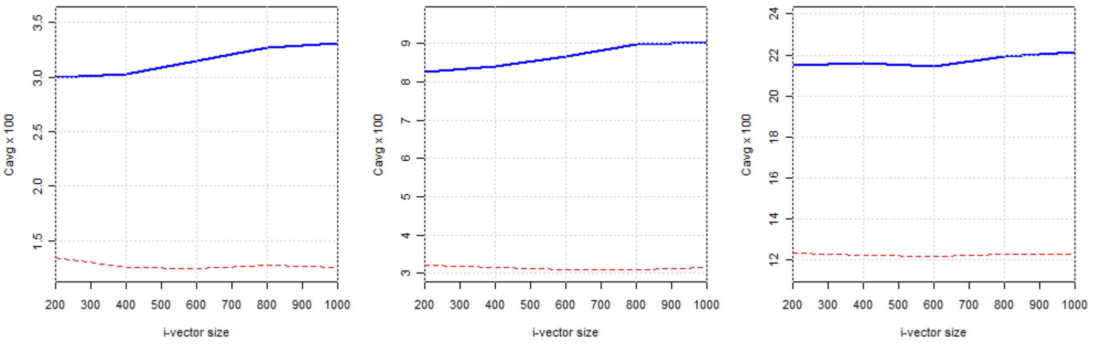

Figure 2. Results for test data on the three different NIST LRE 2009 conditions using the proposed posteriogram trigam counts (solid line) using different i-vector size and the fusion with an acoustic i-vector-based system (dashed line)

The fusion results (fifth row) show that the fused system has a relative improvement of 47.9% for the 30 seconds condition, 37.3% for the 10 seconds condition, and 13.46% for the 3 seconds conditions in comparison with the acoustic system (fourth row). We can also compare these results with the performance of fusing the same acoustic system but with each of the other approaches (i.e. with soft-counts, sixth row, and with using PCA on soft counts, seventh row)

	3 s	*10 s*	*30 s*
1. p-gram i-vectors (600)	**21.45**	**8.66**	**3.15**
2. soft-counts i-vectors (600)	23.70	9.60	3.43
3. soft-counts PCA (1000)	23.17	9.12	3.44
4. Acoustic i-vectors (400)	14.04	4.93	2.40
5. p-gram i-vectors + Ac. i-vectors	**12.15**	**3.09**	**1.25**
6. soft-counts i-vectors + Ac. i-vectors	12.73	3.37	1.39
7. soft-counts PCA + Ac. i-vectors	12.38	3.34	1.39

Table 1. Results (C_{avg}) for three conditions

6. Conclusions and Future Work

In this paper we have presented a feature vector suitable for phonotactic LID. The new system is based on using an easy and robust algorithm that generates *n*-gram counts based on posteriograms. This feature vector has the advantage of reducing the sparseness of the counts matrixes producing a better input to model our phone-based counts. Results on NIST LRE 2009 task by fusing our proposed system with an acoustic i-vector based system provides one of the best results on this task that we know.

As future work we propose the incorporation of an additional mechanism for selecting discriminative n-grams that can help to reduce the size of the input vector [7], as well as using it as scaling factor during the fusion and calibration process.

7. Acknowledgements

This work was done during an internship of Luis Fernando D'Haro at Brno University of Technology funded by the Spanish Ministry of Education through the José Castillejo postdoctoral fellowship. It has also been supported by TIMPANO (TIN2011-28169-C05-03), INAPRA (MICINN, DPI2010-21247-C02-02) and MA2VICMR (Comunidad Autonoma de Madrid, S2009/TIC-1542) projects. In addition, This work was also partly supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. D10PC20015, by Czech Ministry of Education project No. MSM0021630528 and by IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070.

8. References

[1] Reynolds, D. A., Campbell, W. et al, 2007. "Automatic Language Recognition Via Spectral and Token Based Approaches", Springer Handbook of Speech Processing and Communication, Springer-Verlag GMBH, Heidelberg, Germany.

[2] Zissman, M.A., "Comparison of four approaches to automatic language identification of telephone speech," IEEE Trans. Speech&Audio Proc., v. 4, pp. 31-44, 1996.

[3] W. Campbell, F. Richardson, and D. Reynolds, "Language recognition with word lattices and support vector machines," in ICASSP, pp. 15-20, Honolulu, Hawaii, USA, 2007.

[4] J. Goodman, "A Bit of Progress in Language Modeling, Extended Version", Microsoft Research Technical Report MSR-TR-2001-72

[5] Gauvain, J.L., Messaoudi, A, and Schwenk, H. "Language Recognition using Phone Lattices" ICSLP (2004), pp. 1283-1286.

[6] Campbell, W. M., Richardson, F., 2007. "Discriminative Keyword Selection Using Support Vector Machines", Neural Information Processing Systems, Vancouver, BC Canada. Dec. 3-8, 2007.

[7] M. A. Caraballo, L. F. D'Haro, et al. 2010. "A Discriminative Text Categorization Technique for Language Identification built into a PPRLM System". FALA 2010, pp. 193- 196,. Vigo, Spain.

[8] Najim Dehak, Patrick Kenny, et al. 2011. "Front-End Factor Analysis For Speaker Verification". IEEE Transactions on Audio, Speech And Language Processing , Vol. 19, No. 4, May 2011, pp. 788-798.

[9] Soufifar, M. et al. 2011. "iVector approach to phonotactic language recognition". Interspeech, pp. 2913-2916.

[10] Kockmann, et al, 2010. "Prosodic speaker verification using subspace multinomial models with intersession compensation," in Proc. of ICSPL, Makuhari, Chiba, Japan, 2010.

[11] Martínez, D., et al. 2010. "Language Recognition in iVectors Space", Interspeech 2011, Florence, IT, ISCA, 2011, p. 861-864, ISSN 1990-9772

[12] Mikolov et al. 2010. "PCA-based feature extraction for to phonotactic language recognition". Odyssey, pp. 251-255.

[13] D. Povey, Lukas Burget et. al, 2011. "The Subspace Gaussian Mixture Model– a Structured Model for Speech Recognition", Computer Speech and Language, 25(2), pp. 404-439.

[14] Jančík, Z. et al., "Data Selection and Calibration Issues in Automatic Language Recognition - Investigation with BUT-AGNITIO NIST LRE 2009 System", Odyssey 2010 - The Speaker and Language Recognition Workshop, Brno, CZ.