ANALYZING THE EFFECT OF CHANNEL MISMATCH ON THE SRI LANGUAGE RECOGNITION EVALUATION 2015 SYSTEM

Mitchell McLaren[1], *Diego Castan*[1], *Luciana Ferrer*[12]

[1]Speech Technology and Research Laboratory, SRI International, California, USA
[2] Departamento de Computación, FCEN, Universidad de Buenos Aires and CONICET, Argentina
{mitch,dcastan}@speech.sri.com, lferrer@dc.uba.ar

ABSTRACT

We present the work done by our group for the 2015 language recognition evaluation (LRE) organized by the National Institute of Standards and Technology (NIST). The focus of this evaluation was the development of language recognition systems for clusters of closely related languages using training data released by NIST. This training data contained a highly imbalanced sample from the languages of interest. The SRI team submitted several systems to LRE'15. Major components included (1) bottleneck features extracted from Deep Neural Networks (DNNs) trained to predict English senones, with multiple DNNs trained using a variety of acoustic features; (2) data-driven Discrete Cosine Transform (DCT) contextualization of features for traditional Universal Background Model (UBM) i-vector extraction and for input to a DNN for bottleneck feature extraction; (3) adaptive Gaussian backend scoring; (4) a newly developed multi-resolution neural network backend; and (5) cluster-specific N-way fusion of scores. We compare results on our development dataset with those on the evaluation data and find significantly different conclusions about which techniques were useful for each dataset. This difference was due mostly to a large unexpected mismatch in acoustic conditions between the two datasets. We provide a post-evaluation analysis that reveals that the successful approaches for this evaluation included the use of bottleneck features, and a well-defined development dataset appropriate for mismatched conditions.

***Index Terms*—** Language Recognition, Bottleneck Features, Deep Neural Networks, Mismatched conditions.

1. INTRODUCTION

The 2015 NIST LRE focused on the development of language recognition systems for closely related languages using a highly imbalanced training set [1]. The training set's contents range from 20 minutes of speech from one language to orders of magnitude more data for others. The training data was provided by NIST, and participants were restricted by the core condition from using any other data for training, in contrast to previous LREs in which participants were allowed to use any publicly available data for development. Furthermore, unknown to participants prior to the release of results, the evaluation data was highly mismatched to the data provided by NIST for development. In contrast, previous evaluations involved evaluation data that had been well matched to that used by most groups for training and development (composed mostly of data from previous LREs). Another distinctive aspect of LRE'15 was that the 20 target languages were split across 6 clusters. The performance metric was an average of the performance within each cluster. This average allowed, in principle, for the development of 6 completely separate systems that targeted the languages in each cluster.

Most prevalent in recent language recognition literature is the use of bottleneck (BN) features extracted from a deep neural network (DNN) trained to discriminate tied tri-phone states, or senones [2, 3]. These features replace the traditional acoustic features such as Mel Frequency Cepstral Coefficients (MFCC) in an i-vector pipeline. The scoring backend for the resulting i-vectors is often based on a Gaussian Backend (GB), or Neural Network (NN), to classify the language. Alternate approaches for language recognition rely on phone modeling, such as PPRLM [4, 5]. Given that the provided training data for the fixed condition of LRE'15 included phone alignments only for the English SWB1 corpus, these alternative methods were less suitable for this evaluation. Therefore, we chose to focus on only bottleneck i-vectors.

This article describes the systems submitted to LRE'15 by the team from SRI's Speech Technology and Research (STAR) laboratory, and describes additional analysis that highlights key approaches that provide some robustness to the severe mismatch between the evaluation and the development data. Major components of the SRI submission included (1) bottleneck features [6] extracted from DNNs trained to predict English senones, with multiple DNNs trained using a variety of acoustic features; (2) data-driven Discrete Cosine Transform (DCT) contextualization of features [7] for traditional Universal Background Model (UBM) i-vector extraction and for input to a DNN for bottleneck feature extraction; (3) adaptive Gaussian backend scoring [8]; (4) a newly developed multi-resolution neural network backend; (5) cluster-specific N-way fusion of scores; and (6) cluster-specific conversion of scores from likelihoods to likelihood ratios.

2. DEVELOPMENT WITH LIMITED AND IMBALANCED DATA

Considerable effort was put into the construction of a development set from the provided fixed training dataset. Table 1 details the languages, channels and number of audio files in this set. The first step involved splitting the data into training and development sets, with 20% of the audio files for each language used for development and the rest for training. The proportion assigned for development was increased when necessary to ensure a minimum of 10 audio files per language for development. Care was taken to balance channel exposure in both train and dev splits, and more specifically, in the two

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-15-C-0037. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

Table 1. The data available for each of the 20 languages for system training and development under the LRE'15 fixed training condition after removal of duplicates and incorrectly labeled audio. File counts are separated by channel; conversational telephone speech (CTS) and broadcast narrowband speech (BNBS). Note CTS audio with stereo channels are considered as two audio files.

		Channel	
Cluster	**Language**	**CTS**	**BNBS**
ara	ara-acm	420	-
ara	ara-apc	450	-
ara	ara-arb	-	394
ara	ara-ary	414	-
ara	ara-arz	439	-
eng	eng-gbr	-	32
eng	eng-sas	52	344
eng	eng-usg	427	-
fre	fre-hat	-	321
fre	fre-waf	34	-
ibe	por-brz	2	43
ibe	spa-car	119	-
ibe	spa-eur	38	-
ibe	spa-lac	29	-
qsl	qsl-pol	248	234
qsl	qsl-rus	167	301
zho	zho-cdo	41	-
zho	zho-cmn	438	-
zho	zho-wuu	45	-
zho	zho-yue	23	-

splits of development used for cross-validation in calibration/fusion experiments. Following the tradition of previous LRE cycles, we assumed that mismatch of development and evaluation conditions would not be a major factor. As detailed later, this was evidently an incorrect assumption. Consequently, we chose not to make use of the provided Switchboard 1 and Switchboard 2 data in development (with the exception of Switchboard 1 for DNN training), based on the assumption that this data would not be observed in the evaluation data since the complete datasets were delivered for training.

Both training and development splits were processed to produce cuts of audio containing at least 3 sec of speech. All efforts were made to cut segments between non-speech regions as detailed in the evaluation plan [1]. We devised a random speech distribution generator from which the target cut durations were determined. This generator was more biased toward 3-7 sec cuts, since we observed very few errors for long duration samples in the dev set. A plot of the distribution of speech duration is given in Figure 1. This bias is evident when contrasting with the speech duration distribution of the unseen evaluation data in the same figure.

Given that part of the task of LRE'15 was to devise a suitable development set, we analyzed the data for (a) incorrect labels and (b) duplicate files in the training dataset. To help automate the process of identifying incorrect labels, we trained a system to classify all 20 target languages using all provided data, then tested on the same data using a maximum of 30 sec of speech from each file. For each target language, we listened to the lowest-scoring files. We identified five incorrect labels in the eng-gbr dataset and used these for only unsupervised training of the UBM and i-vector subspace. Duplicates in BNBS audio were also located using a windowed match filter on the waveforms. A total of 37 files contained a duplicate with 95% or more overlap in content. All duplicates were removed from the pool of available training data to reduce bias under the limited training conditions. Consequently, the lowest resource language from a duration perspective, eng-gbr, had just 32 original, unique, correctly labeled files of less than 40 seconds each on which to develop. These files were split into 22 for training and 10 for dev.

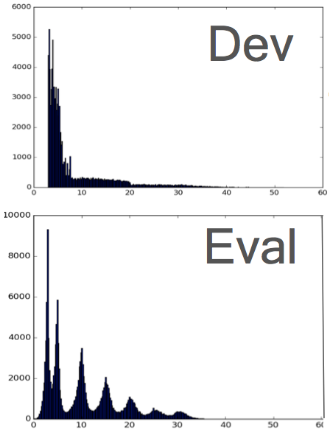

Fig. 1. Distributions of speech durations of the development data and the unseen evaluation data.

3. FEATURES

Our submissions were based entirely on the fusion of different i-vector pipelines from both traditional acoustic features and bottleneck features. Each set of i-vectors was processed with multiple backend scorers to provide a pool of scores from which a final fusion was selected. Given the cluster-specific metric of LRE'15, we adopted a cluster-specific selection of scores for fusion. Using cluster-specific fusion (detailed more in Section 6), we expected that different features would be more suitable than others, depending on the language cluster under evaluation. For this reason, an ensemble of features was produced to populate the score fusion pool. Acoustic features were used as input to traditional i-vector extraction pipelines and to DNN or CNN bottleneck feature extractors prior to an i-vector extraction pipeline. Acoustic features included:

PNCC	Power Normalized Cepstral Coefficients [9]
MHEC	Mean Hilbert Envelope Coefficients [10]
LMS	Log Mel Spectra (filter bank energies)
GCC	Gammatone Cepstral Coefficients
PV	Pitch and voicing

GCC were extracted using the same algorithms as used for PNCC, but log compression, rather than power compression, was applied to the cepstrum. Pitch and voicing estimates (PV) were extracted using the open-source Kaldi package [11].

3.1. Feature contextualization

Each of the acoustic features were contextualized using four different methods: shifted-deltas cepstrum (SDC), deltas and double deltas (D+DD), rank-DCT [7], and pca-DCT [7]. The data-driven methods of rank-DCT and pca-DCT contextualization first involve producing a 2D-DCT matrix for each frame of development speech.

This matrix is sub-selected to remove the first column which represents the mean of cepstral features over a window, and remove the second half of the remaining columns. For rank-DCT, these subsampled 2D-DCT matrices have each of their coefficient indices ranked by their average rank position over the development set. Then, the top 60 coefficient indices are used in the final feature extraction process. PCA-DCT, on the other hand, learns a transformation matrix from the vectorized 2D-DCT coefficients of the speech frames from the development set to retain as much of the speech variability as possible in 60 dimensions. The PCA transform is learned on the training data portion of our development set (as defined in Section 2). The resulting features are then expected to contain the most relevant dimensions in terms of speech content for this specific data.

3.2. DNN/CNN Bottleneck Features

We extracted bottleneck features from several DNNs and CNNs [2, 6, 12] for our submission. The bottleneck features are given by the values of the nodes in a bottleneck layer in a trained DNN at each time frame. The bottleneck layer, a hidden layer in the DNN, has reduced dimensions relative to the other layers (80 nodes compared to 1200 in our systems). For the SRI submissions, both CNNs and DNNs were trained to discriminate between 3021 senones based on alignments generated using a GMM-HMM ASR systems. The networks are trained using the SWB1 dictionary and corresponding dataset provided by NIST for the evaluation. The number of senones was determined by a coarse sweep of senone values from around 800 to 4000 using a DNN bottleneck i-vector pipeline with a Gaussian Backend (GB). DNNs were trained to have 5 hidden layers. CNNs had 4 hidden layers preceded by a convolutional layer of 200 filters of height 8 and width equal to the context size, with max pooling of 3.

Our submission consisted of bottleneck features extracted from a selection of two CNNs and five DNNs. The networks were trained using contextualized acoustic features described earlier in this section, and are detailed below. DNN input features were contextualized (after optionally adding deltas or DCT coefficients) using seven frames on either side of a frame. The bottleneck features from these DNNs/CNNs were then used as input to an i-vector extraction pipeline.

CNN LMS+PV	Log Mel Spectra + pitch + voicing
CNN LMS	Log Mel Spectra
DNN LMSpcadct	Log Mel Spectra with pca-DCT contextualization
DNN MFCCdd	MFCC with D+DD
DNN MFCCdd+PV	MFCC+D+DD + pitch + voicing
DNN PNCCdd	PNCC with D+DD
DNN MHECdd	MHEC with D+DD

4. I-VECTOR EXTRACTION AND SAD MODULES

All subsystems were based on 2048-Gaussian Universal Background Models with diagonal covariance and 400-D i-vector subspaces [13]. Speech activity detection (SAD) for the computation of statistics to train the i-vector extractors was performed using a GMM-based system as in [14]. It should be noted that training data for the SAD model was not constrained under the 'fixed' training conditions. The SAD model was trained on data from the PRISM dataset [15] and additional in-house music data representing different genres of hold music. The model was based on 13D MFCCs with appended deltas and double deltas, and three 128 Gaussian GMMs representing speech, non-speech, and music. A median filter was used to smooth speech likelihood ratios before applying a threshold of 0.

In contrast, all i-vectors (i.e., training, dev, and test audio) were extracted using a DNN-based SAD. This SAD system showed consistent gains over the GMM-based SAD system on our development set. Unfortunately, since this finding was made towards the end of the development cycle, we did not have time to retrain the i-vector extractors to use this SAD method. The DNN-based SAD was trained on the same data, with the addition of music samples. The music samples were created by adding clean speech samples to music-only samples at different SNR levels. The system uses 20-dimensional MFCC features, which were normalized by waveform to have zero mean and a standard deviation of 1 over each dimension, and to be concatenated over a window of 31 frames. The resulting feature vector is input to a DNN with two hidden layers of sizes 500 and 100. The output layer of the DNN consists of two nodes trained to predict the posteriors for the speech and non-speech classes. These posteriors are converted into likelihood ratios using Bayes rule (assuming a prior of 0.5), and smoothed over a window of 41 frames and thresholded at a value of -0.5 to get the final speech regions.

5. BACKEND SCORING

All i-vectors were processed with 4 different scoring backends and the i-vectors from BN features were additionally processed with a multi-resolution NN backend. Details of each backend are given below:

- **Gaussian Backend (GB)**: Traditional GB with shared covariance, and additional class-weighting as in [3] to normalize for class imbalance in the training data. The model was trained using training chunks with 6 or more seconds of speech.
- **Adaptive Gaussian Backend - Top-N (AGBtopn)**: Initially developed in [16], this model creates a test-dependent GB by comparing the test i-vector to the candidate i-vectors per language and using the top-N (e.g., 500) training i-vectors for the dynamically determined GB. To cope with imbalance in this work, the top proportion (20%) of training i-vectors per language was used, instead of fixed top-N. A minimum of 50 i-vectors was maintained to ensure that mean estimates from low-resource languages were not too noisy. The model was trained using training chunks with 6 or more seconds of speech.
- **Adaptive Gaussian Backend - Support Vector Machine (AGBsvm)**: Also first developed in [16] was an SVM extension of AGB. Rather than select the top-N based on Euclidean distance, an SVM was trained to discriminate the test i-vector against the i-vectors of a language, and the resulting support vectors were then used for the mean estimate in the AGB for that language. For the AGBsvm backend, we limited the training chunks to those with more than 15 sec of speech content. We also applied a top-10% reduction of background segments for each test and for each language before training each SVM to significantly reduce computation at no cost to performance.
- **Neural Network (NN)**: A NN backend trained from exhaustive 'chunks' of speech audio (around 8sec with 50% overlap) obtained high performance in the DARPA RATS program [17]. We also found this backend to be very competitive

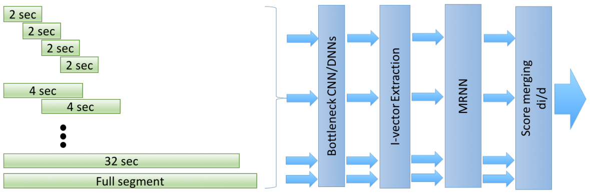

Fig. 2. Multi-resolution Neural Network (MRNN) backend in which the test audio is used to produce multiple i-vectors of exhaustive sampling of speech durations. The final score is produced as a duration-weighted average of the test scores from each i-vector.

on the LRE'15 development set defined in Section 2. The NN backends were trained to discriminate all 20 languages with 250 hidden nodes using numerous chunked i-vectors as defined for our training dataset. Pre-processing using mean and variance normalization was based on the training dataset, and all training i-vectors were used in training (representing speech segments as low as, and dominated by, 3 sec chunks).

- **Multi-resolution Neural Network (MRNN)**: This backend was developed during our LRE'15 development phase; this paper represents the first publication of the technique. The motivation behind this backend was to improve performance by leveraging the short spoken queues that differentiate close language pairs. Consequently, this backend was applied only to bottleneck features, as these contain rich phonetic information when i-vectors are extracted from short speech samples [18]. The MRNN is trained in the same way as the NN backend. However, training segments were exhaustively chunked at each of 2, 4, 8, and 16 second windows with 50% overlap. It should be noted that this different method of training provided marginal gains to the NN backend but was not employed in it due to time constraints. The major performance benefit to the MRNN for LRE'15 comes from the 'chunking' of test samples at multiple durations including 2, 4, 8, 16, and 32 second windows with 50% overlap plus one i-vector from the full speech sample (see Figure 2). Each test chunk is then evaluated by the NN and scores are merged using a weighted average in which each score is weighted by $\frac{d_i}{d}$ where d is the duration of speech in the full test segment, and d_i is the duration of speech that went into the test chunk i. This average weight has the effect of normalizing for the dominant shorter chunks from test files, but also allows for more confident decisions (ideally due to more distinct phonetic content of the detected language/dialect) on short segments to be realized. Similar to the NN backend, the MRNN backends were trained to classify the 20 target languages and included 250 hidden nodes.

6. SCORE-LEVEL FUSION AND CALIBRATION

Unless otherwise stated, each input feature was processed using each backend to generate a set of scores on the development set. This pool of scores (62 sets in total) were candidates in score fusion. Fusion was done using multi-class linear regression with cross-validation (xval) of the development set. We used up to 6-way score-level fusion in our submissions. Exhaustive exploration of the candidate score sets was not feasible, so the following approach was taken:

- Conduct exhaustive 2-way xval fusion.
- Select the best 2-way fusions for each cluster and the average metric (7 in total). Iterate the following until the desired N-way fusion is obtained:
- Conduct exhaustive N-way xval fusion including the 7 selected (N-1)-way fusions.
- Select the best N-way fusions for each cluster and the average metric.

For development, the fusion was trained and applied in a xval scenario. The test scores, however, were fused using a calibration model trained on all scores from the development set.

Rather than exhaustively show the candidate systems for fusion, the final subsystems selected for each of the SRI submissions is detailed in Table 2. Details regarding the objective and motivation behind these submissions are given in Section 8.

Table 2. The subsystems and number of times selected for use in each of SRI's submissions to LRE'15. Subsystems are denoted $[\text{feature}]\,[\text{contextualization}] - [\text{backend}]$. Submissions SRI_01 and SRI_03 may contain more than one instance of the same subsystem due to the use of cluster-specific 5 or 6-way fusion. The type BNiv and iv denote i-vector systems based on bottleneck or acoustic features, respectively, CNNBNiv being bottleneck features from a CNN instead of a DNN. Note that SRI_02 and SRI_04 have the same composition, and that they differ in terms of subsequent score normalization.

		Submission SRI_**			
Subsystem	**Type**	**01**	**02**	**03**	**04**
LMS-GB	CNNBNiv			1	
LMS-NN	CNNBNiv	3	1		1
LMS-MRNN	CNNBNiv	3			
LMSPV-GB	CNNBNiv			3	
LMSPV-NN	CNNBNiv	3	1		1
LMSPV-MRNN	CNNBNiv	1			
LMSpcadct-GB	BNiv			1	
LMSpcadct-AGBtopn	BNiv	1			
LMSpcadct-NN	BNiv	1			
MFCCdd-GB	BNiv			3	
MFCCPVdd-GB	BNiv	1	1	6	1
MFCCPVdd-AGBtopn	BNiv	1			
MFCCPVdd-NN	BNiv	2			
MHECdd-GB	BNiv	2	1	3	1
MHECdd-MRNN	BNiv	1			
PNCCdd-GB	BNiv	1		2	
PNCCdd-NN	BNiv	1			
GCCrankdct-GB	iv			2	
GCCrankdct-AGBsvm	iv	2	1		1
LMSrankdct-GB	iv			2	
LMSrankdct-AGBtopn	iv	1			
LMSrankdct-NN	iv	1			
MHECrankdct-GB	iv	1		5	
MHECrankdct-AGBsvm	iv	1			
PNCCrankdct-GB	iv	1			
PNCCrankdct-AGBsvm	iv	1			
PNCCsdc-GB	iv	1		2	
PNCCsdc-NN	iv		1		1

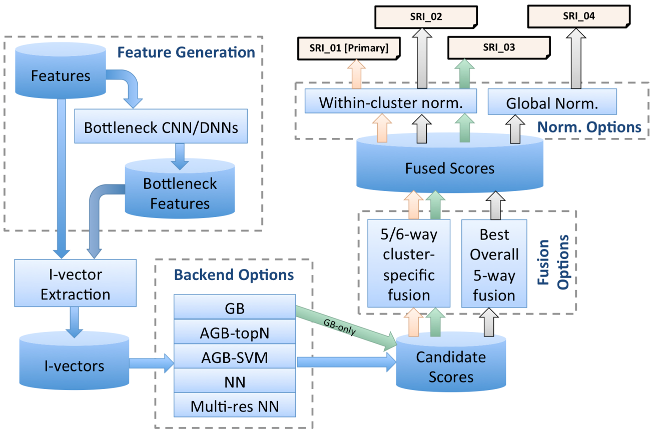

Fig. 3. Flow diagram of data from features through to SRI submissions 1–4 for the fixed training condition of LRE'15. Note that the non-blue arrows denoted branching of processing for particular submissions.

7. CONVERSION OF SCORES TO DETECTION LLRS

The scores coming out of the different backends are calibrated with multi-class logistic regression. The resulting scores are then converted into detection LLRs [19]. This conversion can be done globally or on a per-cluster basis. The first method of 'global' conversion considered all 20 target languages. The alternate method was 'within-cluster' conversion in which only scores from the target languages in the same cluster were considered. One caveat to within-cluster normalization is that comparison of across-cluster scores will be invalidated. This process did, however, greatly assist in optimizing the target within-cluster metrics of our LRE'15 development set.

8. SUBMISSIONS

We submitted a primary system and three contrastive systems for the limited training set condition of LRE'15. Figure 3 provides a concise flow diagram to differentiate submission through use of colored arrows. Details of the primary and how each contrastive system differed are given below:

SRI_01 Our **primary submission** consisted of cluster-specific selection of the 5-way score-level fusions from all input features and applicable backends. Cluster-specific selection attempted to minimize the performance metric for that cluster. Each of these fusions were performed based on all 20 target languages. Only the scores corresponding to the target cluster were extracted from each individual fusion and consolidated as test scores. Within-cluster score normalization was then applied to test scores.

SRI_02 In contrast to SRI_01, this system selected a *single* 6-way fusion of scores as opposed to cluster-specific selection. Within-cluster score normalization was still applied.

SRI_03 This system was the same pipeline as SRI_01, with the exception that backend scoring was restricted only to GB classifiers and 6-way cluster-specific fusion was employed instead of 5-way cluster specific fusion. Within-cluster score normalization was applied. The aim of this system was to quantify the contribution of the non-traditional GB backends to the primary submission.

SRI_04 This system was the same as SRI_02 except global score normalization across all 20 target languages was applied as opposed to within-cluster score normalization in SRI_02.

9. EVALUATION RESULTS

This section presents the results of the submitted systems and the subsystems described in previous sections over the development and the evaluation datasets.

Figure 4 shows the average metric of the subsystems that form part of the four submissions described in Section 8. The subsystems are ranked by the performance over the development dataset. Several conclusions can be extracted from this figure. First, the performance of the subsystems with BN features is generally better than those systems with traditional, non-BN features. These results demonstrate the power of i-vectors from BN features to reflect the spoken language of the audio. Also, the CNN-BN features are better than the DNN-BN features: the four best subsystems are based on CNNs. This may happen because the CNNs are able to highlight the correlation between senones in the time domain that can aid discrimination of different languages. Finally, the two best subsystems use the MRNN proposed in Section 5.

Figure 5 compares the primary and the contrastive systems over the development and the evaluation datasets where an at least five-fold increase in error on the eval set was observed. This significant difference in the range of performance between the dev and the eval results is indicative of the mismatch between the datasets. Furthermore, the relative gain among the submissions does not align between datasets. While the SRI_01 has a 35% gain over the SRI_04 on the development set, this translated to a 2% gain on the evaluation data.

To better determine whether the problem of mismatch was a result of overfitting through fusion or mismatched conditions, we analyzed the performance of each subsystem on the evaluation data. Figure 4 provides this comparison of the subsystems. Here, we can observe that using the single best system on the development set (CNNBNIV-KLMS-MRNN) would have provided similar performance to the 5-way cluster-specific fusion used in SRI_01. Further, the single best subsystem was BNIV-KLMSPV-GB, which offered performance better than any submission. Specifically, while SRI_01 gave a 35% relative gain over the best single subsystem during development, it reflects an 8% loss over the best single subsystem on eval. This indicates that fusion of subsystems provided us with no benefit in the mismatched conditions of LRE'15. Additionally, we can see that our DNN-based bottleneck features were more robust under this mismatch than bottleneck features extracted from CNNs. One trend that did carry over from development results was that BN features were, for the most part, more robust than non-BN features.

10. POST-EVALUATION ANALYSIS

In LRE'15, each team had to devise their own development setup. This section presents some post-evaluation analysis on different key factors that were used across teams, such as the effect of the chunking the data to train the backend systems, the split of the data between train and dev, and some variations in algorithms used by teams in the LRE'15. Finally, an analysis was performed on the sensitivity due to the mismatch in the different modules of the pipeline to help direct future research to addressing the issue of mismatch for language recognition.

We ran the analysis experiments using a single bottleneck i-vector system based on MFCCdd input features to the DNN and a

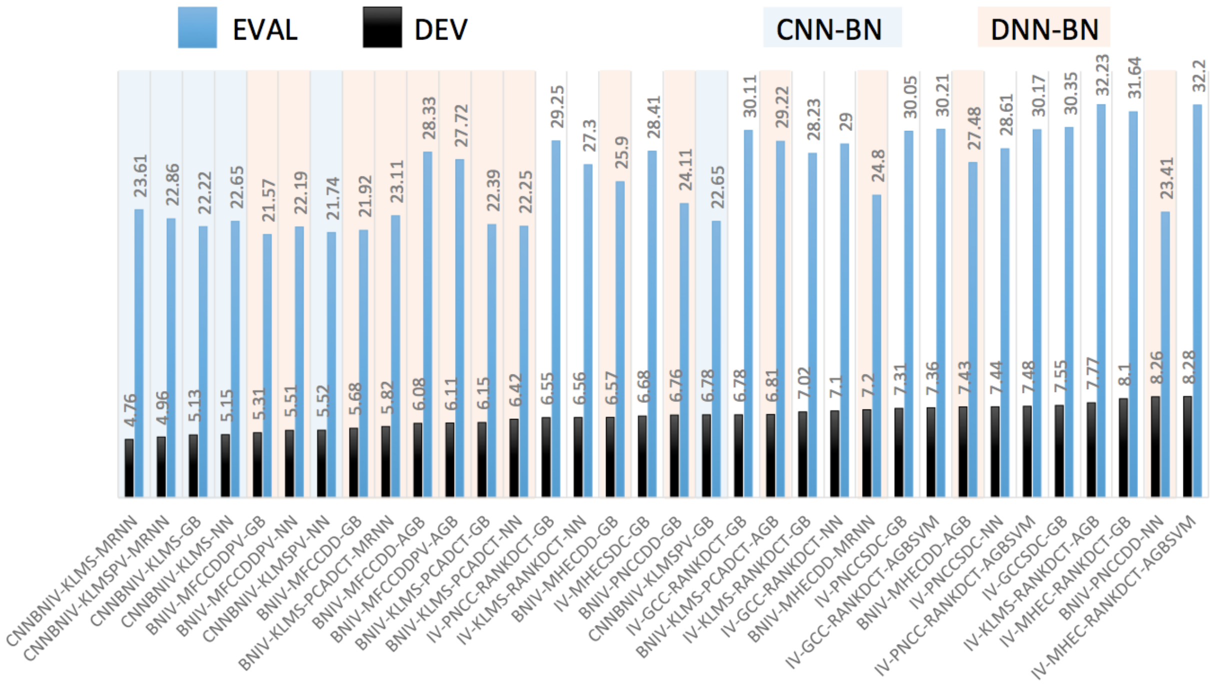

Fig. 4. Subsystems results on both development and evaluation data, ranked by the performance over the development dataset with subsystems based on bottleneck (BN) features being distinguished from traditional, non-BN features. The subsystem naming convention follows ivmodel, feature, contextualization and backend scorer with the ivmodel being CNN bottleneck (CNNBNIV), DNN bottleneck (BNIV), non-bottleneck (IV).

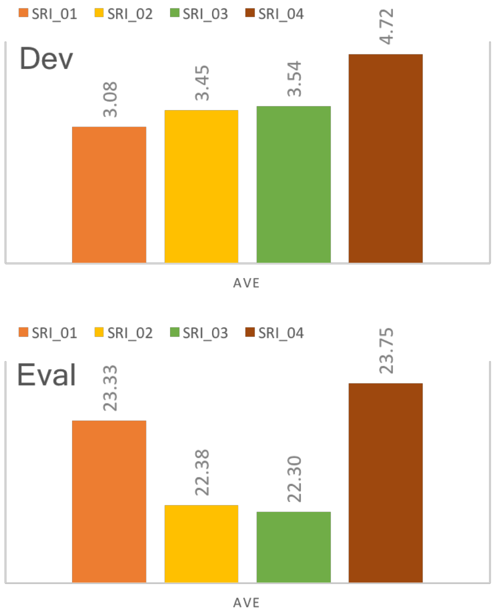

Fig. 5. Results of the submitted systems in the development and the evaluation datasets. Note that the scale of the plot in the evaluation results has been increased 5 fold indicating the degree of mismatch between development and evaluation datasets.

Table 3. Analyzing the effect of dividing limited training data between train and dev partitions on the LRE'15 evaluation results. Several algorithmic differences from our submissions are also provided indicating means of additional robustness.

Train/Dev	Chunking	System	Eval AvgCdet
80/20	Yes	SRI_01	23.3
80/20	Yes	BNiv MFCCdd GB	21.9
60/40	Yes	BNiv MFCCdd GB	21.0
All/All	Yes	BNiv MFCCdd GB	19.7
All/All	No	BNiv MFCCdd GB	18.7
All/All	No	BNiv MFCCdd LWC	19.8
All/All	No	BNiv MFCCdd LWC (cluster-spec. UBM)	18.3
All/All	No	BNiv MFCCdd GB (cluster-spec. UBM)	17.6

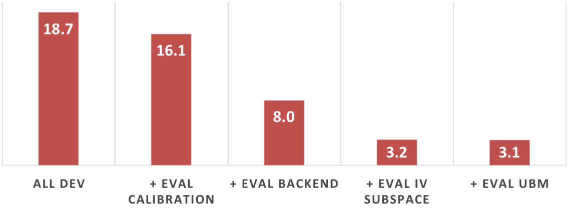

Fig. 6. Effect of incrementally using eval data deeper in the pipeline

GB classifier. This system achieved 5.6% of performance in the dev set and 21.9% in the eval set. Table 3 shows different configurations with this single system to highlight important aspects for LRE'15. The first two rows of the table compare the primary submitted system with the single subsystem used for the post-evaluation analysis, and motivate the use of a single subsystem for post-evaluation analysis. There was some variation across teams as to how the provided data was split between train and development. We analyzed the effect of this by shifting from an 80%/20% train/dev split to 60%/40% and observed an improvement, putatively from the additional dev scores and variation used for system calibration. With this insight, we also trained and calibrated a system using all the available data (that is, train was also dev). The forth row in the table indicates the considerable effect this had on the mismatched evaluation data, with the benefit coming from better use of limited training data for LRE'15. A comparison of the fourth and fifth rows shows that chunking of training files for the GB actually reduced generalization of the system. 'No chunking' refers to the use of a single i-vector per original audio file for GB training, irrespective of speech duration. This benefit may have been due to the dominance of shorter segments in the chunking approach. Overall, the better use of data (all for training and dev, and not chunking) resulted in a relative improvement of a 15% with respect our original subsystem. This indicates that the handling of limited training and development data under mismatched evaluation conditions was a major factor in LRE'15.

The second section of Table 3 shows the results of our single subsystem using promising approaches from other teams in LRE'15. Instead of using GB as a backend system, several teams used Linear Discriminant Analysis (LDA) to reduce the dimensionality of the data to 20 dimensions. This was followed by Within Class Covariance Normalization (WCCN) and cosine distance scoring against an average i-vector for each target language. This approach is denoted as LWC. The alternate approach involved training six i-vector extractors with each having a cluster-specific UBM that was trained on the training data of that cluster only. Despite the increase in computation, this approach provided additional robustness for the evaluation conditions offering and AvgCdet of 17.6. Finally, both techniques were combined using LDA-WCCN with cosine distance as a backend with the UBM cluster-specific strategy, however the LWC backend did not provide gains over GB.

Perhaps the most consistent trend observed in the results presented is that of mismatch between development and evaluation data. We aim now to shed light on the areas of mismatch sensitivity in different parts of the i-vector pipeline for language recognition. For this purpose, we took the single bottleneck i-vector system and incrementally retrained different parts of the pipeline using the evaluation data via a cross-validation approach. Specifically, the evaluation data was split based on unique original IDs and the longest version of each file used to result in a total of around 8700 files. Figure 6 illustrates these results. Firstly, retraining calibration parameters on eval data offered only a 14% improvement (comparing the first and second bars). A major drop in error, a 50% relative reduction, was obtained by taking the eval data deeper into the pipeline and retraining the GB. Furthermore, retraining the i-vector extractors (multiple, due to cross-validation) with eval data gave an additional 60% relative reduction, while re-training the UBM did not improve significantly on this. From this figure we can conclude that both the GB and i-vector extractor were the most sensitive modules of the i-vector pipeline for our LRE'15 submissions. Future research will aim to address these sensitivities to both mismatched training and evaluation data, and limited training data.

11. CONCLUSIONS

This paper has presented the SRI systems submitted to LRE'15. These systems were based on the i-vector paradigm with the best subsystems for mismatched conditions being based on bottleneck features extracted from a DNN. Additionally we proposed a multi-resolution neural network (MRNN) backend which provided the best results on our development set. Results on our development dataset were compared with those on the evaluation data, where considerable mismatch was evident. We provided an analysis on this dataset that reveals the key elements of this evaluation included the use of bottleneck features, a well-defined development dataset appropriate for mismatched conditions, with some additional benefit from advanced algorithms. The paper has shown that the approaches based on a fusion of different subsystems and the chunking of the data to train the systems did not work as expected under the conditions of limited training data and mismatch between training and evaluation data. On the other hand, the robustness of bottleneck features to such conditions was exemplified.

Future work is expected to focus on coping with mismatch in the context of language recognition. Based on the experiments in this study, reducing the sensitivity of the i-vector extractor and backend scorer to mismatch is expected to play an important role in this regard.

12. ACKNOWLEDGMENTS

The authors would like to thank Vikram Mitra, Chris Bartels, and Colleen Richey of the STAR lab for providing the necessary tools to produce alignments and DNNs from scratch for the SRI LRE'15 submission, as well as valuable advice in terms of narrowing the scope of parameters to tune for DNN/CNN. Thanks also goes to MIT, BUT and JHU members for sharing their development lists and/or system descriptions that facilitated the analysis in this article.

13. REFERENCES

[1] *The 2015 NIST language recognition evaluation plan*, 2015, `http://www.nist.gov/itl/iad/mig/upload/LRE15_EvalPlan_v23.pdf`.

[2] Y. Lei, L. Ferrer, A. Lawson, M. McLaren, and N. Scheffer, "Application of convolutional neural networks to language identification in noisy conditions," in *Proc. Speaker Odyssey*, 2014.

[3] L. Ferrer, Y. Lei, M. McLaren, and N. Scheffer, "Study of senone-based deep neural network approaches for spoken language recognition," *IEEE Transactions on Acoustics Speech and Signal Processing*, 2015, Accepted for publication.

[4] P. Matejka, P. Schwarz, J. Cernocky, and P. Chytil, "Phonotactic language identification using high-quality phoneme recognition," in *Proc Interspeech*, 2005.

[5] W. Shen, W. Campbell, T. Gleason, D. Reynolds, and E. Singer, "Experiments with lattice-based PPRLM language identification," in *Proc. Odyssey*, 2006.

[6] Y. Song, B. Jiang, Y. Bao, S. Wei, and L. Dai, "i-Vector representation based on bottleneck features for language identification," *Electronics Letters*, vol. 49, no. 24, pp. 1569–1570, 2013.

[7] M. McLaren and Y. Lei, "Improved speaker recognition using DCT coefficients as features," in *Proc. ICASSP*, 2015.

[8] M. McLaren, A. Lawson, Y. Lei, and N. Scheffer, "Adaptive gaussian backend for robust language identification.," in *Proc. Interspeech*, 2013, pp. 84–88.

[9] C. Kim and R. Stern, "Power-normalized cepstral coefficients (PNCC) for robust speech recognition," in *Proc. ICASSP*, 2012, pp. 4101–4104.

[10] O. Sadjadi, T. Hasan, and J. Hansen, "Mean hilbert envelope coefficients (MHEC) for robust speaker recognition.," in *Proc. Interspeech*, 2012.

[11] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, Georg Stemmer, and Karel Vesely, "The kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[12] P. Matejka, L. Zhang, T. Ng, S.H. Mallidi, O. Glembek, J. Ma, and B. Zhang, "Neural network bottleneck features for language identification," in *Proc. Speaker Odyssey*, 2014.

[13] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. on Speech and Audio Processing*, vol. 19, pp. 788–798, 2011.

[14] L. Ferrer, M. McLaren, N. Scheffer, Y. Lei, M. Graciarena, and V. Mitra, "A noise-robust system for NIST 2012 speaker recognition evaluation," in *Proc. Interpseech*, 2013.

[15] L. Ferrer, H. Bratt, L. Burget, H. Cernocky, O. Glembek, M. Graciarena, A. Lawson, Y. Lei, P. Matejka, O. Plchot, and Scheffer N., "Promoting robustness for speaker modeling in the community: The PRISM evaluation set," in *Proc. NIST 2011 Workshop*, 2011.

[16] M. McLaren, N. Scheffer, L. Ferrer, and Y. Lei, "Effective use of DCTs for contextualizing features for speaker recognition," in *Proc. ICASSP*, 2014.

[17] A. Lawson, M. McLaren, Y. Lei, V. Mitra, N. Scheffer, L. Ferrer, and M. Graciarena, "Improving language identification robustness to highly channel-degraded speech through multiple system fusion," in *Proc. Interspeech*, 2013.

[18] M. McLaren, L. Ferrer, and A. Lawson, "Exploring the role of phonetic bottleneck features for speaker and language recognition," in *Proc. ICASSP*, 2016.

[19] N. Brummer and D. Van Leeuwen, "On calibration of language recognition scores," in *Odyssey: Speaker and Language Recognition Workshop*, 2006, pp. 1–8.