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

Log-linear System Combination Using Structured Support Vector Machines

J. Yang, A. Ragni, M. J. F. Gales and K. M. Knill

Department of Engineering, University of Cambridge, Cambridge, UK
{jy308,ar527,mjfg,kate.knill}@eng.cam.ac.uk

Abstract

Building high accuracy speech recognition systems with limited language resources is a highly challenging task. Although the use of multi-language data for acoustic models yields improvements, performance is often unsatisfactory with highly limited acoustic training data. In these situations, it is possible to consider using multiple well trained acoustic models and combine the system outputs together. Unfortunately, the computational cost associated with these approaches is high as multiple decoding runs are required. To address this problem, this paper examines schemes based on log-linear score combination. This has a number of advantages over standard combination schemes. Even with limited acoustic training data, it is possible to train, for example, phone-specific combination weights, allowing detailed relationships between the available well trained models to be obtained. To ensure robust parameter estimation, this paper casts log-linear score combination into a structured support vector machine (SSVM) learning task. This yields a method to train model parameters with good generalisation properties. Here the SSVM feature space is a set of scores from well-trained individual systems. The SSVM approach is compared to lattice rescoring and confusion network combination using language packs released within the IARPA Babel program.

Index Terms: system combination, structured support vector machines, speech recognition, keyword spotting

1. Introduction

Automatic speech recognition (ASR) systems generally require training on large amounts of data to achieve high accuracy. However, sufficient data cannot normally be guaranteed for many low resource languages. Automated approaches can be used to increase the amount of training data, such as synthesising speech with the known transcriptions [1] and using untranscribed audio in semi-supervised training [2]. An alternative solution to this problem is to use data from other languages to train a multilingual deep neural network (DNN) [3, 4] to produce more robust bottleneck features. Although these data augmentation schemes yield improvements [5, 6], performance is often unsatisfactory. Thus, a combination of multiple well trained systems might be preferred.

In general system combination approaches can be divided into two distinct groups, i.e. hypothesis combination and log-likelihood score combination. Recogniser output voting error reduction (ROVER) [7] and confusion network combination (CNC) [8] are typical examples of combining the hypotheses generated by different recognisers. In these approaches, different hypotheses to be combined are generated from separate decoding runs on various systems. It is computationally expensive to run multiple passes of decoding, and the cost is even higher when speech recognition is an intermediate process, such as in keyword spotting (KWS) [9]. Thus, more efficient approaches based on log-likelihood score combination might be preferred. Here, the log-likelihoods from different systems are combined, and only a single decoding run is required.

Joint decoding [10] is a typical example of log-likelihood score combination, where the state (frame level) log-likelihoods of tandem [11] and hybrid [4] systems are linearly combined. The systems to be combined share the same hidden Markov model (HMM) topology and the combination weights are typically manually set. This makes joint decoding highly constrained. An alternative to combining frame level log-likelihoods is to combine segment level HMM log-likelihoods. This relaxes the frame level Markov assumption to the segment level, and synchronises systems at the phone (or word) level rather than frame. This relaxation also allows long-span dependency within the segments to be captured.

In [12], structured discriminative models are trained using the feature space based on phone log-likelihoods with the same context but different central phone generated by tandem and hybrid systems. Small gains were observed from using additional log-likelihoods extracted from the same models. [13] examines combination of hybrid and tandem systems with log-linear models, and applies learnt phone-specific combination weights to frame level joint decoding, achieving a small performance gain. [14] discusses model combination at sentence level, using system-specific combination weights. A more general framework was introduced by [15], where systems are combined at word level, and the word-specific combination weights are trained with the minimum Bayes risk (MBR) criterion. Another approach was investigated in [16]. However, these approaches are still limited as some words in decoding may not appear in training, especially in low resource language tasks.

The log-likelihood score combination approach is investigated in this paper. This approach is cast into a structured support vector machine (SSVM) [17, 18, 19] learning task to robustly estimate phone-specific combination weights. In this work, a more meaningful feature space is used, which is based on phone log-likelihoods from multiple systems, rather than using extra phone log-likelihoods with the same context extracted from a single system. This paper also discusses assumptions applied in typical combination approaches, such as frame level and segment level lattice rescoring, and investigate what impact these assumptions have on system combination gains. To assess the impact of different approaches on a downstream task, KWS performance is also examined. All experiments are car-

This work was supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense U.S. Army Research Laboratory (DoD/ARL) contract number W911NF-12-C-0012. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

http://dx.doi.org/10.21437/Interspeech.2016-377

ried out on the highly challenging IARPA Babel evaluation task. The paper is organised as follows. In section 2 different commonly used combination approaches and the generation of complementary systems are discussed. The SSVM is introduced in section 3. Finally, experimental results and conclusions are presented in sections 4 and 5.

2. System Combination

In standard decoding, the Viterbi algorithm is employed to find the best hypothesis $\hat{W}$. Given an utterance $\boldsymbol{O}$, the decoding process can be described as:

$$\hat{W} = \arg\max_{W} \left\{ p(\boldsymbol{O}|W)P(W)^{\alpha} \right\} \tag{1}$$

where $p(\boldsymbol{O}|W)$ and $P(W)$ are the likelihood and probability given by the acoustic (AM) and language (LM) models. α is the LM scale factor. In hypothesis combination approaches, such as ROVER [7], multiple decoding runs are required which is computationally expensive. Alternatively, a single set of lattices generated by one system can be rescored by using other systems. Then the hypothesis combination approaches can be applied to the lattices rescored by different systems. This type of approach can significantly reduce the computational overhead as the search space (or hypotheses) is given by the lattice. The rescoring process can be described as:

$$\hat{W} = \arg\max_{W\in\mathbb{L}} \left\{ p(\boldsymbol{O}|W)P(W)^{\alpha} \right\} \tag{2}$$

where $W \in \mathbb{L}$ denotes a possible hypothesis given by a determinised lattice. $p(\boldsymbol{O}|W)$ and $P(W)$ are the likelihood and probability given by AM and LM, respectively, which might differ from the ones used to generate the lattice. This approach is referred to as *frame level lattice rescoring* in this paper.

A far more efficient way to rescore the lattice is to fix both the search space and phone segmentations, referred to as *segment level lattice rescoring*. It can be expressed as:

$$(\hat{W}, \hat{\boldsymbol{\rho}}) = \arg\max_{W,\boldsymbol{\rho}\in\mathbb{L}} \left\{ \left(\prod_{i=1}^{|\boldsymbol{\rho}|} p(\boldsymbol{O}_{(i)}|w_i) \right) P(W)^{\alpha} \right\} \tag{3}$$

where $W, \boldsymbol{\rho} \in \mathbb{L}$ denotes a possible hypothesis and the corresponding segmentation given by a lattice, and $p(\boldsymbol{O}_{(i)}|w_i)$ is the likelihood (corresponding to segment $\boldsymbol{O}_{(i)}$ with triphone label[1] w_i) given by an AM, which might differ from the one generating the lattice. $|\boldsymbol{\rho}|$ is the number of segments. By using logarithms, equation (3) can be rewritten as:

$$(\hat{W}, \hat{\boldsymbol{\rho}}) = \arg\max_{W,\boldsymbol{\rho}\in\mathbb{L}} \left\{ \begin{bmatrix} 1 \\ \alpha \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \sum_{i=1}^{|\boldsymbol{\rho}|} \log p(\boldsymbol{O}_{(i)}|w_i) \\ \log P(W) \end{bmatrix} \right\} \tag{4}$$

This is a linear combination of log-likelihoods from a single system. Before introducing the SSVM approach, the possible ways of generating complementary systems (that make different errors) will be discussed in the following subsection.

2.1. Complementary System Generation

In system combination, it is assumed that the systems to be combined complement each other. The most commonly used approach to generating complementary systems is simply to train a number of independent systems with different acoustic modelling techniques. These individual systems might use different front-ends, segmentations, dictionaries or decision trees [20, 21]. Figure 1 illustrates the framework of the tandem and stacked hybrid systems used in this work. In this framework, complementary systems might be generated, for example, by using different (unilingual or multilingual) data to train the bottleneck DNN, employing different input feature types, using different supervisions to train the transforms, semi-supervised AM training, or using different structures or activation functions for the bottleneck or hybrid DNNs. All these approaches lead to potentially complementary systems, but there is no theoretical guarantee, hence a number of experiments must be performed to select the optimal combination [20]. However, standard approaches to assess how complementary individual systems are would involve multiple decoding runs which are time consuming. Alternatively, complementarity can be efficiently evaluated by the combination approaches based on lattice rescoring with different AMs. This is examined in the experimental section.

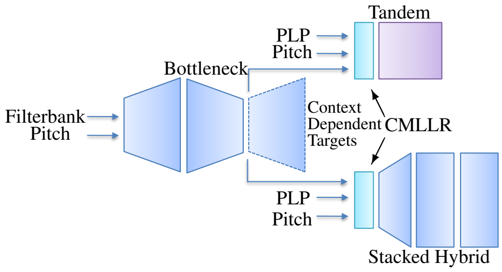

Figure 1: Tandem and stacked hybrid systems.

3. Structured Support Vector Machines

In its basic form, a support vector machine (SVM) is a linear classifier [22]. To classify observation sequences $\boldsymbol{O}$ into one of many possible sentences W with a SVM, the simplest option is to map them jointly into a fixed dimensional representation $\Phi(\boldsymbol{O}, W)$. Unfortunately extracting fixed dimensional features from variable-length observation sequences and modelling the vast, unstructured, mostly unseen space of possible sentences is non-trivial [23]. Instead, a structured assumption is usually imposed on the label space making sentences to be variable-length sequences of finite vocabulary units such as words or phones. The variable-length issue is then delegated to those units by aligning them with observations to yield alignment $\boldsymbol{\rho}$ dependent feature vectors $\Phi(\boldsymbol{O}, W, \boldsymbol{\rho})$. This serves the basis of structured discriminative models including SSVMs. Classification is performed by solving a semi-Markov inference problem [24]:

$$(\hat{W}, \hat{\boldsymbol{\rho}}) = \arg\max_{W,\boldsymbol{\rho}} \left\{ \boldsymbol{\alpha}^{\mathsf{T}} \Phi(\boldsymbol{O}, W, \boldsymbol{\rho}) \right\} \tag{5}$$

Efficient inference can be performed if the search space in equation (5) is constrained to fewer hypotheses encoded compactly in a lattice [25, 26]. Such lattices can be efficiently generated using standard HMM-based approaches [27]. It is simple to notice that the inference problem in equation (5) or its lattice based approximation includes equation (4) as a subproblem. This relationship will be explored in the rest of this section.

The key concept of SVMs is a margin defined as the distance between the closest correct and incorrect class example [22]. For SSVMs this can be defined by:

$$\begin{aligned} &\mathcal{M}\big(W_n, \boldsymbol{\rho}_n, W, \boldsymbol{\rho}; \boldsymbol{\alpha}, \boldsymbol{O}_n\big) \\ &\quad = \boldsymbol{\alpha}^{\mathsf{T}} \Phi\big(\boldsymbol{O}_n, W_n, \boldsymbol{\rho}_n\big) - \boldsymbol{\alpha}^{\mathsf{T}} \Phi\big(\boldsymbol{O}_n, W, \boldsymbol{\rho}\big) \end{aligned} \tag{6}$$

[1] The following discussion on phonemic systems also applies to graphemic systems.

where $(W_n, \boldsymbol{\rho}_n)$ and $(W, \boldsymbol{\rho})$ are correct and incorrect alignments for observation sequence $\boldsymbol{O}_n$. For a sequence classification task it is important to take into account loss, $\mathcal{L}(W_n, W)$, as different alignments make different number of transcription errors. Training then consists of minimising the largest violation of the loss-augmented margin [17]:

$$\mathcal{F}_{\text{LM}}(\boldsymbol{\alpha}) = -\log p(\boldsymbol{\alpha}) + \sum_{n=1}^{N} \left[\max_{W \neq W_n} \Big\{ \mathcal{L}(W_n, W) - \mathcal{M}\big(W_n, \boldsymbol{\rho}_n, W, \boldsymbol{\rho}; \boldsymbol{\alpha}, \boldsymbol{O}_n\big) \Big\} \right]_+ \quad (7)$$

where N is the number of training examples, and $[\cdot]_+$ is the hinge loss. $W \neq W_n$ denotes any possible hypothesis that differs from the reference W_n[2]. $p(\boldsymbol{\alpha}) = \mathcal{N}(\boldsymbol{\mu}_\alpha, C\boldsymbol{I}) \propto \exp\big(-\frac{1}{C}||\boldsymbol{\alpha} - \boldsymbol{\mu}_\alpha||^2\big)$ is the Gaussian prior with mean $\boldsymbol{\mu}_\alpha$ and diagonal covariance $C\boldsymbol{I}$. The objective function in (7) can be efficiently solved by using the cutting plane algorithm [28]. In this work, the segmentations corresponding to the references are the most likely segmentations given by the HMMs. An alternative approach to obtain optimal segmentations by using discriminative models is discussed in [17].

There are a number of possible forms for alignment dependent feature vectors $\Phi\big(\boldsymbol{O}, W, \boldsymbol{\rho}\big)$. Typically they consist of observation and transition features [26]. Observation features are extracted from variable-length observation sequences associated with units whereas transition features are extracted on transitions from one unit to another. One simple form is [29]:

$$\Phi\big(\boldsymbol{O}, W, \boldsymbol{\rho}\big) = \begin{bmatrix} \sum_{i=1}^{|\boldsymbol{\rho}|} \delta(w_i, v_1) \boldsymbol{\phi}\big(\boldsymbol{O}_{(i)}, w_i\big) \\ \vdots \\ \sum_{i=1}^{|\boldsymbol{\rho}|} \delta(w_i, v_L) \boldsymbol{\phi}\big(\boldsymbol{O}_{(i)}, w_i\big) \\ \log P(W) \end{bmatrix} \quad (8)$$

where $\{v_l\}_{l=1}^{L}$ denotes all possible phone units in the dictionary. $\delta(\cdot)$ is the Kronecker delta[3]. $\boldsymbol{\phi}(\boldsymbol{O}_{(i)}, w_i)$ is the observation feature vector for segment $\boldsymbol{O}_{(i)}$. In general, this feature vector can be produced by using any approach capable of mapping variable length observation sequences to fixed length. It can be a first and higher order observation statistics traditionally used with frame level models [30, 31]. Other examples include score spaces [32, 33, 34] and event detectors [26, 16]. A simplest example is given in (4), where the mapping for a segment $\boldsymbol{O}_{(i)}$ with label w_i is a log-likelihood, namely $\boldsymbol{\phi}(\boldsymbol{O}_{(i)}, w_i) = \big[\log p(\boldsymbol{O}_{(i)}|w_i)\big]$. This work employs a more general form, which consists of log-likelihoods from multiple AMs, rather than a single one. Let K be the number of AMs, the observation feature vector $\boldsymbol{\phi}(\boldsymbol{O}_{(i)}, w_i)$ and the corresponding phone-specific weights can be described as:

$$\boldsymbol{\phi}(\boldsymbol{O}_{(i)}, w_i) = \begin{bmatrix} \log p_1\big(\boldsymbol{O}_{(i)}, w_i\big) \\ \vdots \\ \log p_K\big(\boldsymbol{O}_{(i)}, w_i\big) \end{bmatrix}, \ \boldsymbol{\alpha}_{w_i} = \begin{bmatrix} \alpha_{w_i}^1 \\ \vdots \\ \alpha_{w_i}^K \end{bmatrix} \quad (9)$$

By setting the weights corresponding to an individual system to 1 and others to 0, the SSVM will retrieve the performance of segment level lattice rescoring described in (4). Moreover, by using these manually set weights in the prior, optimal weights can be learnt by using the large margin training criterion (7).

[2]Since segmentations are introduced, hypotheses having a different segmentation to the reference are treated as different to the reference.

[3]When segment label w_i is a triphone and v_l is a monophone, in the delta function the context of w_i is stripped off.

This approach is adopted in the experiments, where the SSVMs are trained based on these priors with strong baselines.

4. Experiments

Different system combination approaches discussed in the previous sections are examined in this section, with ASR and KWS performance presented. Experiments are performed on the Swahili and Javanese full language packs[4] released within the IARPA Babel program, which contain around 40 hours of transcribed conversational telephone speech data for training.

Approach	Decoding #	KWS #
Standard	4	4
Rescoring (frame)	1	4
Rescoring (segment)	1	4
SSVM	1	1

Table 1: Resource requirements of different approaches.

A combination of 4 joint decoding [10] (tandem and hybrid) systems is examined. Table 1 lists the number of decoding and KWS runs for different combination approaches. As discussed in section 2, for the "Standard" combination approach, 4 passes of joint decoding are run to generate 4 sets of hypotheses. Then CNC is used to combine these hypotheses. This approach also requires 4 KWS runs, with the final KWS result obtained by merging the 4 KWS posting lists. For the frame and segment level lattice "Rescoring" approaches, only 1 decoding run is required to generate 1 set of lattices which can then be rescored as discussed in section 2. It is worth noting that rescoring may use different pruning settings to the standard decoding. Similar to the standard approach, 4 KWS runs are performed based on these 4 sets of rescored lattices, and finally the 4 KWS posting lists are merged. In the "SSVM" approach discussed in section 3, only 1 decoding run is required to generate the lattices. Then the lattices are rescored by using the log-likelihoods from different systems (those form the joint features) and the learnt combination weights for the SSVM as described in equation (5). Based on the rescored lattices, only 1 KWS run is required.

4.1. Experiments on Swahili

Table 2 gives the word error rates (WER) on the Swahili dev set. S1, S2, S3 and S4 are joint decoding systems, which combine the tandem and hybrid log-likelihoods at frame level. S1, S2 and S3 differ in the structure of the bottleneck DNN[5]: they use 26, 39 and 62 dimensional bottlenecks respectively. S4 uses 39-d bottleneck and semi-supervised training. Lattices generated by the S1 system are used for rescoring experiments. For example, the SSVM combination weights are trained based on lattices generated with a bigram language model (LM). In decoding, bigram lattices are rescored with a trigram LM and the scores from the systems to be combined. In the first block (from row 1 to 4) of Table 2, all the numbers are confusion network (CN) results. The second block lists the CNC results.

Since S1, S2, S3 and S4 use different model structures and are trained with different training technologies, they may complement each other. Moreover, these 4 joint decoding systems have comparable performance (see "Standard" column), this makes a good combined performance possible, and experiments show that the CNC result (43.5%) is much better than the result of any individual system. For the system combination results,

[4]Swahili IARPA-babel202b-v1.0d, Javanese IARPA-babel402b-v1.0a.

[5]These DNNs are generated by different sites (CUED and RWTH).

System	Standard	Rescoring		SSVM	
		Frame	Segment	Manual	Train
S1	44.7	44.8	45.0	43.8	43.1
S2	45.6	45.3	45.7		
S3	44.6	44.6	45.1		
S4	44.7	44.7	45.5		
CNC	43.5	43.5	43.7	–	–
Time (hrs)	19	7	10	9	9

Table 2: WER performance on Swahili (202) Eval15 dev set.

System	Standard	Rescoring		SSVM	
		Frame	Segment	Manual	Train
S1	0.543	0.543	0.541	0.559	0.558
S2	0.537	0.535	0.539		
S3	0.540	0.543	0.545		
S4	0.544	0.543	0.535		
PM	0.570	0.566	0.561	–	–
Time (hrs)	25	14	15	3	3

Table 3: MTWV performance on Swahili (202) Eval15 dev set.

frame level lattice rescoring achieves the same result (43.5%), indicating that the S1 lattices contain good sets of hypotheses enabling equation (2) to approximate equation (1) accurately. Segment level lattice rescoring gives a worse result (43.7%), suggesting the segmentations from S1 lattices are not optimal for S2, S3 and S4. Since the rescoring approaches only need 1 decoding run, the complementarity of individual systems can be efficiently evaluated by these approaches. For the SSVM, with manually set weights[6], the performance (43.8%) is worse than the CNC result of the "Standard" approach. Phone-specific trained weights yield a 0.7% absolute performance gain in the SSVM to 43.1%, which is better than the "Standard" CNC result 43.5%. As seen in Table 2, the frame level lattice rescoring takes slightly less time than SSVM due to the time consuming generation of log-likelihood scores.

Table 3 gives the KWS performance for the different combination approaches, measured by maximum term weighted value (MTWV) [35]. The "Standard" posting list merge (PM) approach (that merges the posting lists generated by S1, S2, S3 and S4) achieves the best result (0.570). However, this approach is highly inefficient as 4 decoding and 4 KWS runs are required. More efficient approaches based on lattice rescoring can be used, with 1 decoding and 4 KWS runs. Although frame and segment level rescoring yield accurate approximations to frame-level inference in terms of WER, these approaches display sensitivity in terms of KWS. They obtain worse PM results, i.e. 0.566 and 0.561 for the frame and segment levels. These results are better than any result of the individual systems S1, S2, S3 and S4, given in the "Standard" column. Since 4 KWS runs are highly inefficient, the SSVM combination approach can also be adopted, where only 1 KWS run is required. As shown in Table 3, this is $5\times$ faster. Although for both the manually set and trained weights, the performances are worse than the "Standard" PM result, the "SSVM" results (0.559 and 0.558) are much better than the "Standard" results of the individual systems. SSVM training does not help for KWS, as it aims to reduce overall classification errors but gives high deletion error rate. In conclusion, the suggested KWS approach is a SSVM with manually set weights, since, compared with other schemes, this approach is much more efficient and has much better performance than any individual systems.

[6]These system-dependent weights are $(\frac{\boldsymbol{\alpha}}{4}, \frac{\boldsymbol{\alpha}}{4}, \frac{\boldsymbol{\alpha}}{4}, \frac{\boldsymbol{\alpha}}{4})$ for S1, S2, S3 and S4, where $\boldsymbol{\alpha} = (0.25, 1.0)$ corresponding to tandem and hybrid.

System	Standard	SSVM	
		Manual	Train
J1	53.0	52.6	52.4
J2	53.6		
J3	54.6		
J4	59.8		
CNC	52.4	–	–
Time (hrs)	49	13	13

Table 4: WER performance on Javanese (402) dev set.

System	Standard	SSVM	
		Manual	Train
J1	0.451	0.461	0.459
J2	0.446		
J3	0.420		
J4	0.362		
PM	0.463	–	–
Time (hrs)	40	8	8

Table 5: MTWV performance on Javanese (402) dev set.

4.2. Experiments on Javanese

It is time consuming to generate multiple complementary systems using different model structures or modelling techniques. A simple way to create possible complementary systems is to use different bottleneck features, and this approach is examined on the Babel Javanese data. The systems J1, J2, J3 and J4 presented in Table 4 only differ in the bottleneck DNNs, which are trained on different data sets: J4 – unilingual; J3 – 11 languages[7]; J2 – 24 languages[8]; J1 – 24 languages with fine-tuning to Javanese. These individual systems can be relatively easily generated, but the results in Table 4 show that these systems are not good examples of complementary systems. The CNC result (52.4%) is not significantly better than the J1 result (53.0%). These systems have high deletion rate. The SSVM can only match the CNC result. This indicates that it is hard for SSVMs to take advantage of extra features from other systems, when individual systems are not complementary. However, the SSVM with trained weights is the most efficient and effective approach.

For the KWS results shown in Table 5, the SSVM with manually set weights[9], has a comparable result (0.461) to the PM (0.463), but only 1 KWS run is required by the SSVM, and only needs $1/5$ of the time used by the standard approach.

5. Conclusions

This paper has examined the impact of different combination approaches on both the ASR and KWS performance. The proposed SSVM combination approach, that only needs 1 decoding and 1 KWS run, can achieve good ASR and KWS performance very efficiently. However, training the SSVM weights gives worse KWS performance which needs to be investigated further.

[7]Cantonese IARPA-babel101b-v0.4c, Assamese IARPA-babel102b-v0.5a, Bengali IARPA-babel103b-v0.4b, Pashto IARPA-babel104b-v0.4aY, Turkish IARPA-babel105b-v0.4, Tagalog IARPA-babel106-v0.2f, Vietnamese IARPA-babel107b-v0.7, Haitian Creole IARPA-babel201b-v0.2b, Lao IARPA-babel203b-v3.1a, Tamil IARPA-babel204b-v1.1b, Zulu IARPA-babel206b-v0.1d.

[8]Plus Kurmanji Kurdish IARPA-babel205b-v1.0a, Tok Pisin IARPA-babel207b-v1.0b, Cebuano IARPA-babel301b-v2.0b, Kazakh IARPA-babel302b-v1.0a, Telugu IARPA-babel303b-v1.0a, Lithuanian IARPA-babel304b-v1.0b, Swahili IARPA-babel202b-v1.0d, Guarani IARPA-babel305b-v1.0a, Igbo IARPA-babel306b-v2.0c, Amharic IARPA-babel307b-v1.0b, Mongolian IARPA-babel401b-v2.0b, Javanese IARPA-babel402b-v1.0b, Dholuo IARPA-babel403b-v1.0b

[9]Since the individual systems do not have comparable performance, the system-dependent weights are set to $(0.6\boldsymbol{\alpha}, 0.3\boldsymbol{\alpha}, 0.1\boldsymbol{\alpha}, \mathbf{0})$.

6. References

[1] M. J. F. Gales, A. Ragni, H. AlDamarki, and C. Gautier, "Support vector machines for noise robust ASR," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2009, pp. 205–210.

[2] L. Lamel, J.-L. Gauvain, and G. Adda, "Lightly supervised and unsupervised acoustic model training," *Computer Speech & Language*, vol. 16, no. 1, pp. 115–129, 2002.

[3] F. Grézl, M. Karafiát, S. Kontár, and J. Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proceedings of ICASSP*, vol. 4, 2007, pp. IV–757.

[4] G. Hinton, L. Deng, D. Yu, D. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, pp. 2–17, Nov. 2012.

[5] A. Ragni, K. M. Knill, S. P. Rath, and M. J. F. Gales, "Data augmentation for low resource languages." in *Proceedings of Interspeech*, 2014, pp. 810–814.

[6] J. Cui, B. Kingsbury *et al.*, "Multilingual representations for low resource speech recognition and keyword search," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015.

[7] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 1997, pp. 347–354.

[8] G. Evermann and P. Woodland, "Posterior probability decoding, confidence estimation and system combination," in *Proceedings of Speech Transcription Workshop*, vol. 27, 2000.

[9] M. J. F. Gales, K. M. Knill, A. Ragni, and S. P. Rath, "Speech recognition and keyword spotting for low resource languages: Babel project research at CUED," in *Proceedings of International Workshop on Spoken Language Technologies for Under-Resourced Languages*, 2014.

[10] H. Wang, A. Ragni, M. Gales, K. Knill, P. Woodland, and C. Zhang, "Joint decoding of tandem and hybrid systems for improved keyword spotting on low resource languages," in *Proceedings of Interspeech*, 2015.

[11] H. Hermansky, D. W. Ellis, and S. Sharma, "Tandem connectionist feature extraction for conventional HMM systems," in *Proceedings of ICASSP*, vol. 3, 2000, pp. 1635–1638.

[12] R. C. van Dalen, J. Yang, H. Wang, A. Ragni, C. Zhang, and M. J. F. Gales, "Structured discriminative models using deep neural-network features," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015.

[13] J. Yang, C. Zhang, A. Ragni, M. J. F. Gales, and P. C. Woodland, "System combination with log-linear models," in *Proceedings of ICASSP*, 2016.

[14] P. Beyerlein, "Discriminative model combination," in *Proceedings of ICASSP*, vol. 1, 1998, pp. 481–484.

[15] B. Hoffmeister, R. Liang, R. Schlüter, and H. Ney, "Log-linear model combination with word-dependent scaling factors," in *Proceedings of Interspeech*, 2009, pp. 248–251.

[16] G. Zweig, P. Nguyen, D. Van Compernolle, K. Demuynck, L. Atlas, P. Clark, G. Sell, F. Sha, M. Wang, A. Jansen, H. Hermansky, K. Karakos, D. Kintzley, S. Thomas, G. S. V. S. Sivaram, S. Bowman, and J. Kao, "Speech recognition with segmental conditional random fields: final report from the 2010 JHU summer workshop," Tech. Rep. MSR-TR-2010-173, 2010.

[17] S.-X. Zhang, "Structured support vector machines for speech recognition," Ph.D. dissertation, University of Cambridge, 2014.

[18] S. Ravuri, "Hybrid DNN-latent structured SVM acoustic models for continuous speech recognition," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 37–44.

[19] Y.-H. Liao, H.-y. Lee, and L.-s. Lee, "Towards structured deep neural network for automatic speech recognition," in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 137–144.

[20] M. J. F. Gales, D. Y. Kim, P. C. Woodland, H. Y. Chan, D. Mrva, R. Sinha, and S. E. Tranter, "Progress in the CU-HTK broadcast news transcription system," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 5, pp. 1513–1525, 2006.

[21] C. Breslin and M. J. F. Gales, "Complementary system generation using directed decision trees," in *Proceedings of ICASSP*, vol. 4, 2007, pp. IV–337.

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[23] P. Nguyen, G. Heigold, and G. Zweig, "Speech recognition with flat direct models," *IEEE J. STSP*, vol. 4, pp. 994–1006, 2010.

[24] S. Sarawagi and W. W. Cohen, "Semi-Markov conditional random fields for information extraction," in *Advances in Neural Information Processing Systems (NIPS)*, 2004, pp. 1185–1192.

[25] M. I. Layton, "Augmented statistical models for classifying sequence data," Ph.D. dissertation, University of Cambridge, 2006.

[26] G. Zweig and P. Nguyen, "A segmental CRF approach to large vocabulary continuous speech recognition," in *ASRU*, 2009, pp. 152–157.

[27] J. J. Odell, "The use of context in large vocabulary speech recognition," Ph.D. dissertation, University of Cambridge, 1995.

[28] T. Joachims, T. Finley, and C.-N. J. Yu, "Cutting-plane training of structural SVMs," *J. MLR*, vol. 77, pp. 27–59, 2009.

[29] A. Ragni and M. J. F. Gales, "Structured discriminative models for noise robust continuous speech recognition," in *Proceedings of ICASSP*, 2011, pp. 4788–4791.

[30] A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt, "Hidden conditional random fields for phone classification," in *Eurospeech*, 2005, pp. 1117–1120.

[31] H. K. J. Kuo and Y. Gao, "Maximum entropy direct models for speech recognition," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 3, pp. 873–881, 2006.

[32] N. D. Smith, "Using Augmented Statistical Models and Score Spaces for Classification," Ph.D. dissertation, University of Cambridge, 2003.

[33] T. Jaakkola and D. Haussler, "Exploiting generative models in disciminative classifiers," in *Advances in Neural Information Processing Systems (NIPS)*, 1999, pp. 487–493.

[34] M. J. F. Gales, S. Watanabe, and E. Fosler-Lussier, "Structured discriminative models for speech recognition," *IEEE Signal Processing Magazine*, pp. 70–81, 2012.

[35] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proceedings of ACM SIGIR Workshop on Searching Spontaneous Conversational Speech*, vol. 7, 2007, pp. 51–57.