Babelon

Period –End Technical Report

Period of Performance: 5 June 2014 –30 June 2015

<table>
<tr><td>Organizations:</td><td colspan="2">BBN Technologies (BBN)
Brno University of Technology (BUT)
Johns Hopkins University (JHU)
Vocapia Research (LIMSI)
Massachusetts Institute of Technologies (MIT)
North-West University (NWU)</td></tr>
<tr><td>Principal Investigators:</td><td>Dr. John Makhoul
Tel: 617-873-3332
Fax: 617-873-2473
Email: makhoul@bbn.com</td><td>Dr. Stavros Tsakalidis
Tel: 617-873-4976
Fax: 617-873-2473
Email: stavros@bbn.com</td></tr>
<tr><td>Reporting Period:</td><td colspan="2">5 June 2014 – 30 June 2015</td></tr>
</table>

Babel systems at BUT

Similarly to last period before, BUT is producing systems with two different toolkits: one set of systems is created with the HTK/STK/TNet toolkits – referred to as "STK" systems and one set of systems is created with the Kaldi toolkit – referred to as "Kaldi" systems. The STK and Kaldi systems described here are the final systems that were used in the development and surprise language evaluations in March/April/May/June. The details on system development and insights are described in the following sections.

VAD system (multi-lingual)

Since the multi-lingual VAD is used in both the STK and Kaldi systems we describe it before the systems themselves.

During the OP2 period, we have completely re-designed the infrastructure for Voice Activity Detection (VAD). In the previous version, we used language-specific neural network trained with TNet and HMM-decoder to extract the segments. While in new version the network is trained with Kaldi on multi-lingual corpus and the segments are obtained by applying a threshold to smoothed per-frame scores.

Our multi-lingual VAD consists of 2 carefully designed parts: a neural network (NN) which produces per-frame scores and post-processing which creates the segments from the scores.

The NN is trained multi-lingually on all FLP Y1+Y2 train sets (11 languages), which makes 842 hours in total. The NN is relatively small, it has only 277k parameters. The input dimension is 288, while there are 2 hidden layers, each with 400 sigmoid neurons, and the final softmax layer has 2 outputs, corresponding to the classes: speech, non-speech. The training targets were prepared by mapping from the mono-lingual forced-alignments (generated with GMM/HMM model).

The input features for the NN consist of 15 log-Mel filterbank outputs and 3 kaldi-pitch features [Ghahremani 2013]. We apply per-speaker mean and variance normalization estimated on the whole unsegmented recordings. Then we apply frame splicing with 31 frame-long context, where the temporal trajectory of each feature is scaled by a Hamming window and reduced to 16 dimensions by discrete Cosine Transform. The final 288-dimensional features are globally shifted and scaled to have zero mean and unit variance on the NN input.

In the post-processing we take the pre-softmax NN-outputs, convert them to logit-posteriors and apply smoothing which averages over 31 frames. In the next step the “initial” segments are extracted by using a threshold -0.5, to retrieve a little more speech than with the implicit zero threshold. Finally, the initial speech segments are extended by adding 30 frames on both ends, which merges some of the segments. Because it is time-consuming to process too long segments, we re-split those longer than 15 seconds in the middle third at the frame with the lowest score.

With the current version it is very easy to change setup for specific speech tasks by adjusting the threshold and frame-extension. For example in ASR/KWS we typically have longer segments with some silence inside. On the other hand for Speaker identification we need to remove all the silence frames, while it is acceptable if we remove small amount of speech, which would be harmful for ASR. An example of distribution of per-frame scores is in Fig.1, the threshold is set to -0.5.

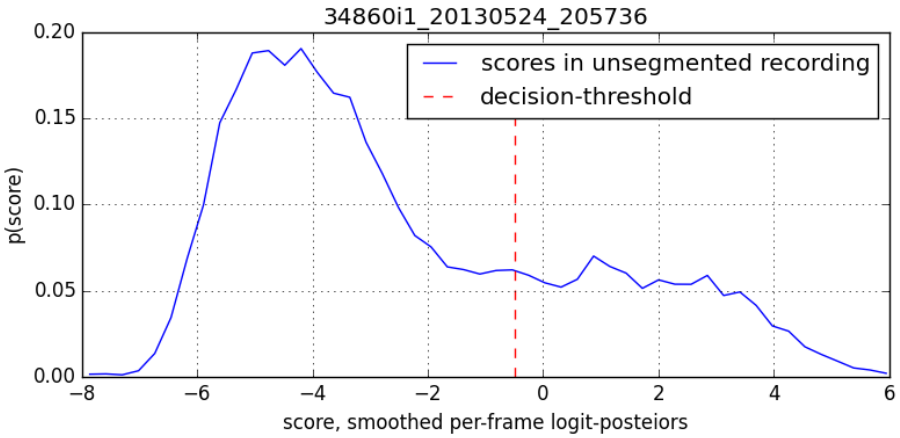

Figure 1: Distribution of scores in unsegmented recording

VAD version	***WER [%]***
Multi-lingual on unseen language	43.8
TokPisin FLP	43.8
TokPisin VLLP	43.9
Cantonese FLP on unseen language	44.1

Table 1: TokPisin VLLP dev results, the multi-lingual VAD generalized to unseen language (TokPisin).

Table 1 shows how the multi-lingual VAD generalizes to a new language. We compare the multi-lingual VAD applied to unseen language (TokPisin) with respect to VADs which are trained on the target language (using FLP 40h or VLLP 3h). The ASR performance is evaluated using TokPisin VLLP system. We see that the multi-lingual VAD works equally well as language-specific VADs. On the other if we use a mono-lingual VAD trained on different language (Cantonese, FLP 138h), the WER performance is worse by 0.3%.

STK systems

These systems were built mainly using the HTK toolkit (http://**htk**.eng.cam.ac.uk/) and STK (http://speech.fit.vutbr.cz/software/hmm-toolkit-stk/) – an HMM modeling toolkit developed at BUT. It provides similar interface and functionality as HTK, while supporting several extensions (e.g. re-estimation of linear transformations MLLT, LDA, HLDA within the training process and use of recognition networks for training).
TNet (http://speech.fit.vutbr.cz/software/neural-network-trainer-tnet) is a fast tool for parallel training of neural networks (NN) for classification, allowing for the NN tricks such as convolutive-bottleneck networks with shared weights [Vesely(2011b)]. We use the STK systems mainly to build discriminative features based on Neural Networks and Region Dependent Transforms – often referred to as "BUT features". The BUT features are based on the concatenation of two feature streams:

1. PLP HLDA (39 dimensional).
2. Stacked Bottle-Neck Neural Network (SBN) [Grezl et al.(2009)] (30 dimensional) which is hierarchical composition of two NN (context-1stageNN and merger-2stageNN) followed by MLLT transform for better de-correlation.

This results in a 69-dimensional feature stream which is further processed by Region Dependent Transforms (RDT) [Zhang et al.(2006)] . The transformation is generated on the feature stream rotated by a single speaker specific CMLLR transform.

The STK speech recognition systems are HMM-based with cross-word tied-states triphones. They were trained from scratch using mix-up maximum likelihood (ML) training. Plain SBN systems (no SAT, RDT) were built for a big part of the analysis, they are referred later as STK BN systems.

In the final system, the SBN architecture was extended by adding:

- Speaker adaptive transform between the first and second stage NN - these 1st stage SAT features were very successfully used as input features for the Kaldi based Deep Neural Net (DNN) system.
- Multilingual fine-tuning – the multilingual NN was fine-tuned into target language domain in two steps using these datasets:
    1. Supervised+unsupervised data – Semi-Supervised Training (SST).
    2. Supervised data.

Stacked Bottle-Neck feature extraction

The SBN structure introduced in [Grezl et al.(2009)] contains two NNs: the BN outputs from the first one are stacked, down-sampled, and taken as an input vector for the second NN. This second NN has again a BN layer, of which the outputs are taken as input features for GMM/HMM recognition system. Our previous study [Vesely(2011a)] has shown that BN neurons with linear activation functions provide better performance.

Mel filter-bank outputs were preprocessed for NN training according to Figure 2. We used 24 Critical-Band Energies (CRBE) with 10 F0-related coefficients. The filter-bank spans frequencies from 64Hz to 3800Hz. The F0-related coefficients consist of F0 and probability of voicing estimated according to [Talkin 1995] and smoothed by dynamic programming, F0 estimates obtained by Snack tool(http://www.speech.kth.se/snack) function *getf0* and seven coefficients of Fundamental Frequency Variations spectrum [Laskowski, et al, 2008]. Conversation-side-based mean was subtraction applied. Next, we stack 11 frames of these features and multiply by Hamming window along the temporal axis. Finally, the temporal trajectories are decorrelated by a DCT transform with 0th to 5th basis.

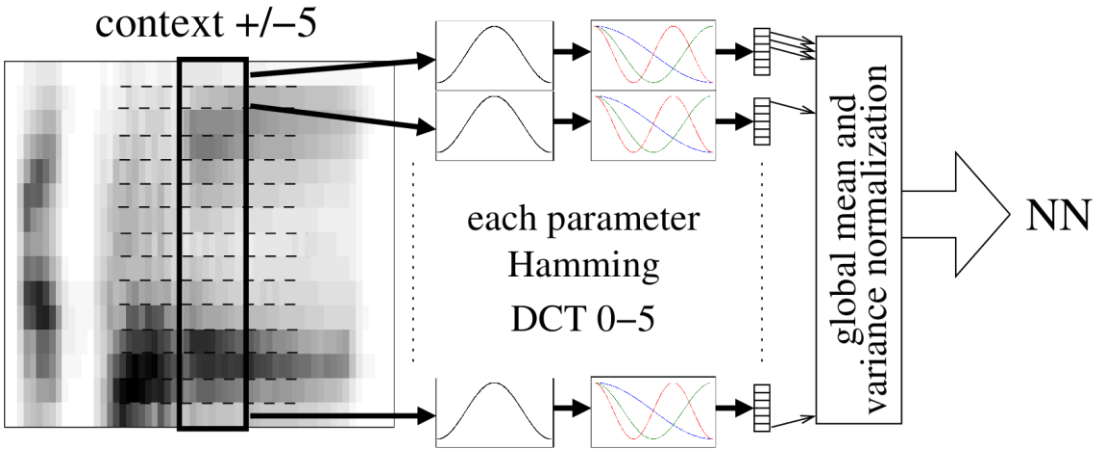

Figure 2: Generating NN input features.

The Stacked Bottle-Neck architecture involves two NNs: the BN outputs from the first one are stacked, down-sampled, and taken as an input vector for the second NN. This second NN has again a BN layer, of which the outputs are taken as input features for GMM/HMM recognition system.

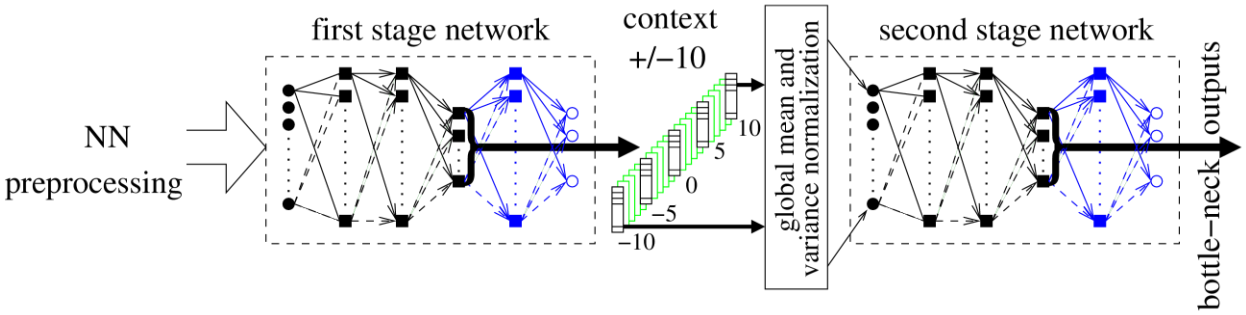

Figure 3: Stacked Bottle-Neck Neural Network.

NN topology

Our experiments questioned the necessity of the hidden layers (HLs) after the BN. Two kinds of SBN hierarchies were trained. The first one following the original description is called *IN-HL-HL-BN-HL-OUT*. In the second case, the hidden layer after the bottle-neck was omitted and the NNs have the following topology: *IN-HL-HL-BN-OUT* with a direct BN-layer -- output-layer connection.

The recognition results using these two variations of DNNs are shown in the first two lines of Table 2. To our surprise, the second version of DNN provided better results than the original structure. Encouraged by these results, a third version of SBNs was trained. It had one more hidden layer before the BN, thus a having topology *IN-HL-HL-HL-BN-OUT*. This version has a similar number of parameters as the original structure. Results using this structure are on the third line of Table 2. For Haitian and Lao, a further improvement was achieved, however, a slight degradation is observed for Zulu.

NN type	Haiti %WER	Lao %WER	Zulu %WER
in-hl-hl-BN-hl-out	65.7	62.4	73.4
in-hl-hl-BN-out	65.0	62.1	73.1
in-hl-hl-hl-BN-out	64.4	61.6	73.3

Table 2: Performance of SBN systems employing DNNs with different topologies.

Multilingual feature extraction

Fully language-independent multilingual feature extraction should generate a better starting point when the system is ported to a new language. Naturally, all of the trainable front-end blocks could be trained in multilingual fashion to maximally use the multilingual resources. The availability of a sufficient amount of data from other BABEL languages and their admission in this year's primary condition motivated us to investigate in building a fully multilingual feature-extraction. As mentioned before, the STK front-end can be decomposed into the following steps:

1. Concatenation of SBN and PLP HLDA:
    - Multilingual version of PLP HLDA was investigated in [karafiat 2012]. We found that the HLDA transform does not need to be trained on the target language, therefore the Tamil HLDA was selected for further experiments.
    - The Multilingual SBN is the consistent focus of our interest.
2. Region Dependent Transforms (RDT) - Initial experiments with multilingual version were presented in [karafiat:ICASSP2012:MultRDT], therefore we further work in this field.
3. Voice Activity Detection (VAD) generates segmentation for Cepstral Mean and Variance Normalization (CMN, CVN).

Multilingual SBN systems

In the previous years of the Babel program [grezl 2014b, grezl 2014c], we already experimented with the training of Multilingual SBN. The following setup was established:

- The "block-softmax" architecture (Figure 4.) - divides the output layer into parts according to each individual language. During the training, only the part of the output layer is activated, which corresponds to the language that the given target belongs to. This approach was successfully used in [Vesely et al.(2012)].
- 2 NNs with 4 hidden layers (1500 neurons each) – the third layer was a BN.
- The context-independent phoneme states were used as training targets in multilingual NN training.

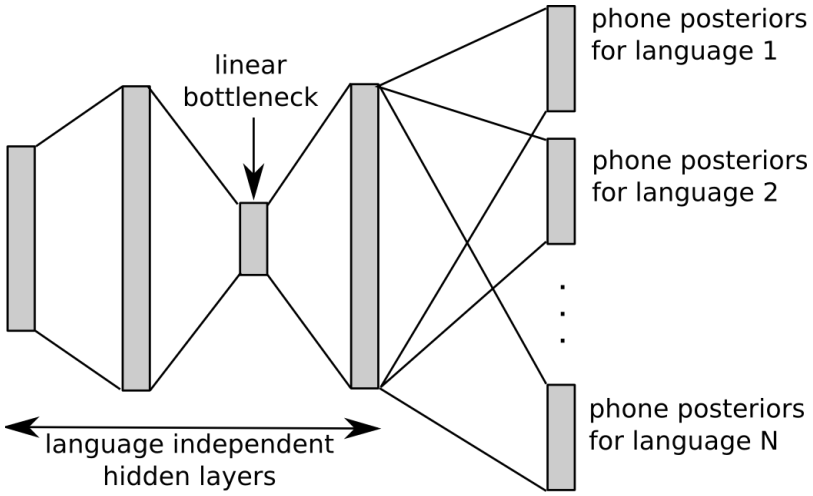

Figure 4: Block-softmax multilingual NN.

This multilingual NN provided the starting point for the final "fine-tuning" to the target language. It is done in two steps:

1. Training of the last layer - the last layer of the multilingual NN was dropped and a new one is initialized randomly with the number of outputs given by the number of tied states in the target language. Only this layer is trained keeping the rest of the NN fixed.

2. Retraining of the whole NN - the remaining layers were released and the whole NN was retrained. The starting learning rate for this phase was set to one tenth of the usual value.

3. The best performing scenario from our previous work [Grezl 2014] was to let both NNs from the SBN hierarchy undergo the same porting process described here. Even if porting the first NN basically changes the inputs to the second one, so that problems with adaptation could be expected, the experiments revealed that the second NN can also adapt to slight changes of the input features.

Multilingual SBN system – porting

Since the topology of DNNs in the target language SBN feature extraction is inherited from the multilingual one, the next step was to train the multilingual DNNs with original topology (*IN-HL-HL-BN-HL-OUT*), and with the best topology (*IN-HL-HL-HL-BN-OUT*) and to evaluate the ported system. The multilingual NN was trained on the year 1 Babel languages.

NN type in final language	Haiti %WER	Lao %WER	Zulu %WER
5lang multiling. NN: in-hl-hl-BN-hl-out			
in-hl-hl-BN-hl-out (baseline)	61.7	57.6	71.1
in-hl-hl-BN-out	61.4	57.1	70.8
5lang multiling. NN: in-hl-hl-hl-BN-hl-out			
in-hl-hl-hl-BN-hl-out	61.5	57.4	70.8
in-hl-hl-hl-BN-out	61.0	56.9	70.8
5lang multiling. NN: in-hl-hl-hl-BN-out			
in-hl-hl-hl-BN-out	64.4	60.9	73.0

Table 3: Multilingual SBN – new versus old topology

The modified DNN topology in the multilingual scenario does not perform as well as expected. According to table 3, a 2-3% degradation is observed when using the multilingual NN without the hidden layer before output layer – it is not well suited for the subsequent porting.

To be able to utilize both positive aspects – having an output layer right after the Bottle-Neck layer for monolingual NNs and having a (large) hidden layer between the Bottle-Neck and output layers - we need to change the porting procedure. The multilingual DNN will be trained with hidden layer between Bottle-Neck and output layer. Then:

1. All layers after bottle-neck will be cut off. A new BN-to-output layer will be initialized randomly and trained, keeping the rest of the NN fixed.
2. The whole network will be retrained as in previous cases.

The modified porting procedure can be found in Table 3. A 0.3-0.7% absolute improvement was reached by adding one more layer in the multilingual NN and by removing the last hidden layer during the porting of the NN into a new language.

Next, we were interested in the effect of adding more data. Therefore, we trained the multilingual NN on 10 languages (year 1 + 2, without Tamil) and applied the new porting approach to Tamil and TokPisin.

NN type in final language	Tamil %WER	TokPisin %WER
5lang multiling. NN: in-hl-hl-hl-BN-hl-out		
in-hl-hl-hl-BN-hl-out	76.9	52.2
in-hl-hl-hl-BN-out	76.5	51.7
10lang multiling. NN: in-hl-hl-hl-BN-hl-out		
in-hl-hl-hl-BN-hl-out	76.0	50.6
in-hl-hl-hl-BN-out	75.9	50.0

Table 4. Porting multilingual SBN - 10 languages.

The direct BN output -- output-layer connection (*IN-HL-HL-HL-BN-OUT*) shows a 0.1-0.6% absolute improvement in comparison to the hidden layer after the BN layer (*IN-HL-HL-HL-BN-HL-OUT*), see Table 4. Using more training data for the multilingual NN gives 0.6-1.7% absolute improvement.

Multilingual vs. monolingual system

According to our previous work on LLP [grezl 2014b], most of the gain from multilingual training disappears when SST was also used. The quality of the initial unsupervised transcripts is significantly influencing the performance of the final system, therefore we were interested in similar experiments in the VLLP condition. The acoustics model of the monolingual system is significantly weaker due to lack of data, therefore the effect of the multilingual system should result in a bigger difference.

System	Seed flat-start features	TokPisin dev WER[%]
No SST, just PLP align	PLP	52.3
1iter. of SST	PLP	48.9
2iter. of SST	PLP	47.9
3iter. of SST	PLP	47.3
4iter. of SST	PLP	47.5
Multi-10L -> SST ->VLLP	Multi-10L	45.4

Table 5: Monolingual vs. multilingual system with SST.

Table 5 shows a 0.4% absolute improvement by building a system seeded from multilingual BN features instead of PLPs. We can see that after the third iteration of SST, the performance of the purely monolingual system saturates. Moreover, the PLP based initialization does not reach the performance of the multilingual based system (2% absolute difference).

Multilingual RDT features

RDT are estimated using gradient-descent algorithm. To obtain shared update statistics, it is enough to sum statistics from language-specific speech recognition systems. Computing derivatives and estimation of the new RDT update follows the standard procedure. It is an iterative procedure, so the updated transformation is cloned to all language-specific systems, new HMM models are re-estimated and new statistics are collected. This process is repeated till convergence is reached.

The RDT were trained on Year 1 (5lang.) or all Y1+Y2 (11) languages. It is applied to the concatenated feature stream (PLPHLDA+MultNN, trained on 10 languages using old topology (*IN-HL-HL-BN-HL-OUT*) in the previous section). The training followed this procedure:

The initial language specific models were estimated by Single-Pass-Retraining (SPR) and the shared transforms were initialized with identity matrices for the central context and with zero for the others.

Language specific gradients were estimated, averaged and the transforms were updated in gradient descent fashion.

Finally, new language specific models were trained. This process was repeated till convergence was reached.

Training data	GMM	WER
Vietnamese	Vietnamese	53.3
Vietnamese	Multilingual – 125G	53.3
Y1	Multilingual – 125G	53.6
Y1	Multilingual – 500G	53.7
Y1	Multilingual– 1000G	53.9
Y1+Y2	Multilingual – 125G	**53.2**

Table 6: Multilingual RDT tested on Vietnamese FLP - one of the training languages.

Table 6 shows the results on one of the training languages, Vietnamese, with various configurations. We see no difference from using multilingual regions instead of language specific ones. A bigger number of regions (richer coverage of the feature space) also do not give any gain, probably due to over-training. More training data achieves a 0.4% gain, but only a tiny improvement is reached over the language-specific system. This outcome was expected as the amount of training data for each language is already high, therefore using information from other languages is not beneficial. However, adding data from more languages naturally covers more variability, therefore it should generalize better to new, previously unseen, languages.

Application of multilingual RDT on unseen languages

Next, we investigated the performance of using the previously trained RDT on Y1+Y2 languages to a new unseen language, where the amount of training data is limited. TokPisin from the Y3 languages was chosen. The initial PLP based HMM system was trained on the Very Limited Language Pack (VLLP), which contains about 3h of data. The input features (PLPHLDA+MultNN) were processed through different RDT configurations that we tested and new models were trained with using SPR.

Training data	GMM	WER
TokPisin VLLP	TokPisin VLLP - 125G	51.2
Tagalog FLP	Tagalog FLP - 125G	52.0
Y1	Multilingual – 125G	50.3
Y1+Y2	Multilingual – 125G	50.2

Table 7: Multilingual RDT on TokPisin

Table 7 shows the dependency of RDT on the target data. 0.8% degradation was observed when the RDT was borrowed from other language. On the other hand, using multilingual RDT presented a 1.7% absolute improvement when it was trained on 5 (Y1) languages and 1.8% on 11 (Y1+Y2) languages.

Therefore, a configuration with 125 regions generated by GMM, and RDT trained on 11 languages was fixed as a standard recipe for further experiments and denoted later as ``MultRDT".

Flat Start training with multilingual RDT features

Next, we were interested in building the TokPisin VLLP HMM system from scratch on various features. The NN based features are trained to learn acoustics clues, therefore straightforward Maximum Likelihood (ML) flat-start models should perform significantly better than traditional spectrum-based features (PLPs). Moreover, the NN-based features should provide a significantly faster convergence with less Gaussians due to the emergence of articulatory clusters.

Input features	WER[%]
PLP	70.0
Multilingual NN Y1only - 5L	53.7
Multilingual NN Y1+Y2 no Tamil - 10L	51.9
NNltilingual NN Y1+Y2 - 11L	50.7
MultRDT	50.0

Table 8: Various flat-start features on TokPisin. Initial ML results.

Table 8 presents a huge 16% absolute improvement over the simple ML VLLP system when multilingual NN based (Y1 - 5languages) features were used. By adding more data into the training, we obtained another significant improvement. Another gain comes from adding Tamil, probably due to its varying recording conditions that were the most variable/difficult across all Babel languages so far. Next, adding multi-lingual RDT gave a 1.9% absolute improvement compared to the MultNN features trained on 10 languages (same NN features were used to RDT input).

Data augmentation

Next, we experimented with artificially generation/cloning the data due to lack of transcribed data in the VLLP condition.

Vocal Tract Length Perturbation

The data was cloned by regenerating of features with various warping factors similarly to experiments proposed by [Cui 2014] and [Jaitly 2013]. Both approaches were compared on simple ML NN based system trained on TokPisin VLLP:

- v1 - Generating of random warping factors (Jaitly 2013)
- v2 – Warping factor shifts (Cui 2014).

System	WER [%]
Baseline	56.6
VTLP v1	56.5
VTLP v2	56.5

Table 9: VTLP on TokPisin VLLP.

Table 9 presents minor 0.1% absolute improvement from VTLP therefore we further focused on data augmentations by waveform modifications.

Noising/Reverberation of data

We generated a parallel corpus by adding various distortions: noises, reverberations, pitch modifications. This allows us to train the VLLP system on more data. Last year on Tamil we found a significant increase in system robustness. All features (PLP, FBANK) were regenerated and the NN training targets were copied from the clean data.

Reverberation

We generated artificial room impulse responses (IR) using the "Room Impulse Response Generator" tool from E. Habets (http://www.audiolabs-erlangen.de/content/05-fau/professor/00-habets/05-software/01-rir-generator/rir_generator.pdf). The tool can model the size of room (3 dimensions), the reflectivity of each wall, the type of microphone, the position of source and microphone, the orientation of the microphone towards the audio source and the number of bounces (reflections) of the signal. Each room model consists of a triplet of IR. One is used for the speech source, one for the inside-room noise, and one for the on-the-wall noise. We randomly set all parameters of the room for each room model. Just the coordinates of the audio source differ for each of the IRs in a-triplet. The constraint was that the distance of speech- and inside-room noise sources from any of the walls should be larger than 0.5~m. On contrary, the distance between a given wall and the on-the-wall noise source should be smaller than 0.5~m. Each dimension of a room was limited to the range 2--5 meters. For a given room model, speech was reverberated (convolution with room IR) using the speech source IR. The noise was reverberated

by randomly chosen noises inside or noises-on-the-wall IR. The speech signal was compensated for the delay caused by the reverberation (to match the timing with the original one).

We used the *fant* tool [Hirsch 2005] to mix the reverberated speech and the reverberated noise at a given SNR.

Cloning the data by using pitch variations

Here we used the “sox” tool to modify the pitch in the recordings. We used the following pitch modification parameters (-300 -200 -100 100 200 300) and created 8 copies of the Swahili database.

Composition of the training set

The training database was cloned 3x with three levels of noise (low, mid, high). The noises were extracted using VAD (non-speech segments) from the recordings of the Year 1+2 languages for the VLLP conditions and from the language specific recordings in the FLP conditions. We also extracted babble noises using the VAD and the BABEL Y1 and Y2 data. Here we overlap 50 speakers per each language.

Three sets of SNR levels for speech and noise mixing were generated. These sets are denoted by ``high SNR" for 25-45dB, ``mid SNR" for 15-25dB and ``low SNR" for 0-15dB. We selected randomly the noise or babbling noise, the SNR level, the room IR and modified the Swahili data.

Each speaker was cloned by 4x random noise/reverb-modifications and 4x random pitch modifications. Therefore, total trainable amount of data was 9x bigger than original size.

Experiments

Complete system (NN feature extraction + GMM) on development languages was trained on noised data. The pitch modifications were used later.

System	Lihuanian WER[%]	Kurdish WER[%]	Cebuano WER[%]	Kazakh WER[%]
Clean	54.0	71.9	61.5	59.1
Noised 4x	52.0	70.2	60.3	56.9

System	Cebuano		Kurdish	
	MTWVnorm[%]	Oracle[%]	MTWVnorm[%]	Oracle[%]
Clean	25.64	36.31	13.76	24.93
Noised 4x	27.58	37.52	16.79	29.43

Table 10: Training on noise: ASR and KWS results

Table 10 shows about 2% WER and 2-3% on MTWV absolute improvement by training on noised data.

The using of noise modifications shows quite promising improvement on all development languages. Consequently we applied this technique on surprise language – Swahili. Here we also investigated the effect of all particular noise/pitch modification on each system part.

The effect of noise on GMM system only was evaluated by generation of multilingual RDT features on distorted wave files. It follows by basic flat-start ML GMM training. The WEB data LM was used.

Training data	Swahili dev WER[%]
Baseline – no noise added	54.7
Original + 4x noise cloning	54.1
Original + 8x noise cloning	53.9
Original + 4xPitch variations	53.2
Original + 4x noise cloning + 4xpitch variations	53.0

Table 11: Flat-start GMMs trained on various waveform modifications.

Table 11 presents impressive 1.5% absolute gain from pitch variations. Adding noise variations shows a slight 0.2% improvement. It seems that pitch modifications are not complementary to pitch modifications.

Next, we trained ML GMMs on clean VLLP data only and built only NN front-end on noises. They were trained in SST manner on VLLP +unsupervised transcripts. Moreover, the data was cloned to one noise specific mirror, so the NN was trained on original data plus one noise variation.

NN training data – noise type	Noise Level		
	Low noise level WER[%]	Middle noise level WER[%]	High noise level WER[%]
Baseline – NN on clean only	53.6		
No reverberation; Babble noise	53.5	53.1	53.4

No reverberation; Background noise	53.2	53.1	52.9
Reverberation; Babble noise	53.2	53.0	53.1
Reverberation; Background noise	52.9	53.1	52.8

Table 12: Effect of noises on front-end level

Table 12 presents smaller improvements from cloning the data on front-end level than on GMM level. Next, it seems that reverberation and high level background noise is beneficial and babble noise is less effective than background noise.

In the same way, we investigated an effect of pitch variations on NN front-end level. Again, the NN was trained on original data and single type of pitch varied clone

Size of pitch change	Swahili dev	
	Slow WER[%]	Fast WER[%]
Original data only	53.6	
1 semitone	54.2	53.6
2 semitone	54.2	53.7
3 semitone	54.0	53.8
4 semitone	54.1	54.0

Table 13: Effect of pitch variations on front-end level

Table 13 shows no improvement from pitch variation in NN training. Therefore it seems that pitch variations are effective only on GMM training.

Similarly to development languages, we built the full GMM system on Swahili VLLP on noised data. The distortions were applied during the NN training and GMM training. NN was trained by fine-tuning from multilingual NN (trained on year 1 + year 2 = 11languages) to SST data which were cloned to various noises.

System	Swahili dev WER[%]
no noise added	50.4
4xnoises	49.4
4x noises + 4xpitches	48.9

Table 14: Final GMM system with training on noises.

Table 14 presents 1% absolute improvement by cloning the data by adding the noise. Next 0.5% improvement can be reached by pitch modifications.

With using last year training recipe we also build DNN hybrid classifier in Kaldi toolkit. It contained pre-training, SST, sMBR…

System	Swahili tune WER[%]	Swahili dev WER[%]
no noise added	53.2	48.5
4x noises	50.7	-

Table 15: Final DNN system trained on noises

Table 15 presents 2.5% absolute improvement by noising the data which is even more than in GMM training. It seems that noising the data is mainly effective on classifier level, which is DNN in this case.

The noising of the data is a cheap technique which can cope insufficient train data issue, like in VLLP. Therefore we were interested in effect of this technique in FullLP condition. The final STK GMM and DNN systems were using the same recipe like last year as the multilingual training was not allowed in FLP condition. The noises were generated in same way like VLLP with difference that only parts from Swahili LP were taken (background noises, speech for babbling).

GMM systems	Swahili dev WER[%]
no noise added	44.1
4x noises	42.0

DNN systems	Swahili dev WER[%]
no SST	41.5
SST	40.8
4x noises – no SST	40.9
No SST, no Noises – Appen dictionary	39.8

Table 16: Noising the data: GMM and DNN systems on FLP conditions.

Table 16 presents 2% absolute improvement from noises on GMM system and 0.6% improvement on DNN. Moreover 0.7% absolute improvement from SST was reached on DNN system.

Next, we investigated the effect of grapheme-based dictionary on system performance. Table 16 shows 1.7% improvement by using original expert based Appen dictionary.

Kaldi systems

Kaldi [Povey et al.(2011)] is a speech recognition toolkit, which has become popular in the research community due to its open spirit and flexibility. It features many traditional state-of-the-art techniques as well as the recent ones, such as exact lattice generation, subspace Gaussian mixture models (SGMMs) with speaker adaptive and discriminative training and Deep Neural Network (DNN) training. A big advantage of Kaldi is that it includes a set of recipes to build state-of-the-art speech recognition systems. These were adapted and extended for the purposes of our experiments.

In the OP2 period, we used Kaldi mainly to build the VLLP systems. The recipe is the following: we **initially build GMM-HMM models** to get tied-state decision trees and transcription forced-alignments. Then we **import multi-lingual features** from the 1st bottleneck of the SBN architecture. And finally we **build a DNN** on top. Note that prior to the import, the multi-lingual features were adapted both to a target language by "fine-tuning" and target speaker by CMLLR transform.

The language models were produced by interpolation of 2 LMs. A first LM trained on VLLP transcripts and a second LM trained on Web-data, the vocabulary of second LM was limited vocabulary to 50k words.

To build the initial GMM-HMM models we use flat-start mixing-up ML, using PLP + Kaldi-pitch features. We add cross-word triphone tied-states (steps "tri1","tri2b"), LDA+MLLT transform on a splice of 9 speech frames (step "tri4b"), and speaker dependent fMLLR transform (step "tri5b"). We used approximately 1700 tied-states, where each state is modelled by 9 Gaussians. Both the LDA+MLLT features and fMLLR features are 40 dimensional. This fMLLR system is later used to produce DNN training targets by forced-alignment, also the tied-state decision trees are re-used for DNN training.

The DNN training recipe features generative RBM pre-training, frame classification training, sequence-discriminative training, and semi-supervised training. A more detailed description will appear later in this document.

Kaldi-GMM input features

Our GMM feature set are PLPs (13 dimensions) concatenated with Kaldi-pitch (3 dimensions). All features (16 dim) were normalized by CMN/CVN and then either deltas/double deltas were applied or we splice 9 frames and estimate LDA+MLLT transform. In the final SAT stage we estimate per speaker fMLLR transforms (40x40 dim):

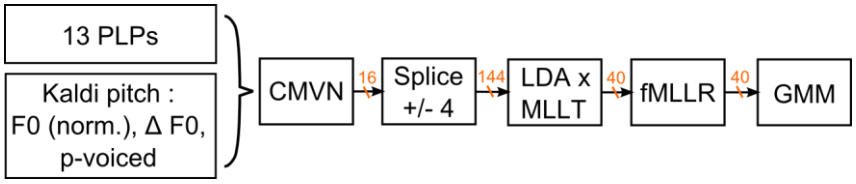

Figure 5: Feature extraction pipeline for Kaldi GMM systems

The Kaldi pitch [Ghahremani 2013] is composed of 3 features: the pitch, delta-pitch and voicing feature. The pitch is smoothed by dynamic programming without making hard-decision about voicing. Further it is normalized by using floating window of 151 frames. The delta-pitch is computed from unnormalized pitch.

Kaldi Deep Neural Network systems

The Deep Neural Networks (DNNs) are very popular in acoustic modeling. The DNNs typically produce triphone tied-state posterior probabilities, and these are used as emission probabilities in HMM decoder. To convert the posteriors into ``log-likelihoods'' we divide them by the tied-state priors. Historically this setup is called as a ``hybrid'' because it is a combination of a neural network and an HMM.

The advantage of DNN over GMMs is that they are trained discriminatively to classify frames, while GMMs are trained generatively. Another good property of a DNN is that the weights specific to tied-states are only in the last layer, while the hidden layers are shared by all the states. Next advantage is that DNN input features are not required to be decorrelated, so we can easily create temporal context by splicing several frames. All these advantages translate into excellent performance of DNN systems.

On the other hand, a performance of a GMM system can be greatly improved when using Bottle-Neck features (called ``tandem systems''), which leads to performance commensurate to DNNs, while giving complementary outputs. Our STK system is an example of a tandem system.

In the first and second year of the Babel program we were both using and developing these training techniques:

- **Generative RBM pre-training**, using the Contrastive divergence algorithm with one step of Gibbs sampling.

- **Mini-batch Stochastic Gradient Descent** optimizing the frame-level cross-entropy, using GMM alignments as training targets. The triphone tree is inherited from the initial GMM system.

- **Sequence discriminative training** optimizing the sMBR criterion - done by Stochastic Gradient Descent with per-utterance updates [Vesely et al.(2013)a]. This aims to maximize the expected frame accuracy within a population of alternative hypotheses represented as a lattice, while the reference path is obtained by forced-alignment.

  In this year there were enhancements to sMBR training : a) the objective function was modified to penalize insertions b) we re-estimate the tied-state priors after last epoch of sMBR training by forwarding the training data and calculating average posterior vector.

- **Semi-supervised training** with frame-weighted SGD training based on per-frame confidences derived from lattice posteriors of states on best path [Vesely et al.(2013)b]. The original recipe from the paper was simplified in the second year by down-scaling the lattice-posteriors which leads to confidences which closely match the ideal confidence, i.e. the probability of a label being correct. The transcribed data are included only once into the train-set, and we no longer needed frame-rejection. As mentioned in the paper, the semi-supervised training is applied only to the per-frame cross-entropy training.

Kaldi-DNN input features (BN)

As mentioned above, we used the multi-lingual STK features on the input to Kaldi DNN. These 80-dimensional features were extracted from the first stage bottleneck, from a network which was previously “fine-tuned” to target language, while we also applied per-speaker CMLLR transforms. On the DNN input we have spliced 11 frames from consecutive interval -5..+5, while we also added 2 extra frames with offsets -10 and +10 w.r.t. the central frame.

Given the success of STK bottleneck features on DNN input from second year, we have been working on producing BN features with Kaldi, and we have successfully replicated the results with Bengali mono-lingual LLP system, see Table 17. In this year, it happened to be crucial to have multi-lingual features for a good VLLP performance, which is why we have again used STK bottleneck features.

DNN systems, Bengali LLP, mono-lingual	dev WER[%]
1xSST on STK BN features (2x SST)	56.8
1xSST on fMLLR features	59.6
1xSST on Kaldi BN features	57.9
1xSST on Kaldi BN features	56.6

Table 17: Replicating the Bengali LLP STK-BN feature result with Kaldi bottlenecks

Table 18 demonstrates the importance of using multi-lingual bottleneck features for obtaining a good performance on VLLP languages. We have observed huge absolute improvement in range 4.2-6.6%, compared to mono-lingual systems which were entirely trained in Kaldi. Both systems were using bottleneck features and semi-supervised training applied to both the BN features and the DNN.

DNN input	Kurdish WER [%]	TokPisin WER [%]	Cebuano WER[%]	Kazakh WER[%]	Telugu WER[%]	Lithuanian WER[%]
Kaldi BN - monolingual SST	76.2	47.0	63.7	64.1	79.7	61.0
STK BN - multilingual SST	70.1	42.8	58.9	58.7	73.1	56.5

Table 18: Importance of using multi-lingual BN features for good VLLP performance

Improvements from Web-data

In table 19 we show the gains coming from 'Web-data' added to language model training. The language models were produced by interpolation of 2 LMs. A first LM trained on VLLP transcripts and a second LM trained on Web-data, the vocabulary of second LM was limited vocabulary to 50k words. We see big differences in the absolute WER improvements across languages from 0.2% (TokPisin) to 6.3% (Lithuanian).

LM corpora	Kurdish WER [%]	TokPisin WER [%]	Cebuano WER[%]	Kazakh WER[%]	Telugu WER[%]	Lithuanian WER[%]
VLLP transcripts	70.1	42.8	58.9	58.7	73.1	56.5
VLLP transcripts + Web-data	69.6	42.6	58.1	55.7	72.4	50.2

Table 19: Improvement from adding Web-data to LM training corpora.

Augmenting data for surprise language

For the VLLP surprise language task, we made experiments with perturbing the input data. We tried separately adding noise and adjusting the pitch of the original speaker. The data augmentation was applied during semi-supervised training where the supervision was generated on clean data. From the results in table 20, we see that noising is very efficient perturbation technique, while the pitch adjustment did not lead to an improvement. This conclusion is contradictory to the observation from STK system.

Perturbation in BN features on DNN input (DNN on hours) VLLP, Swahili	Cross-Entropy WER [%]	sMBR WER [%]	MTWV (FST)
None (73h)	54.2	53.2	0.430
Adding noise (340h)	52.2	50.7	0.456
Pitch adjustment (292h)	54.1	53.2	-

Table 20: Augmenting data for both BN and DNN training by perturbation

Kaldi decoding and Keyword spotting

We use the Kaldi recipe to generate the decoding network, we generate lattices in Kaldi-WFST format, by using DNN acoustic scores (pre-softmax tiedstate scores with log-priors subtracted) while decoding with the HCLG recognition network.

We tuned the acoustic-scale to obtain optimal OracleTWV on tune-set, while we also made sure that the pruning beams are large enough.

In total, we generated 3 types of lattices:

I. Word lattices
II. Phone lattices
III. BBN sub-word lattices

For all the 3 lattice-types, the recognition networks were based on the trigram LMs. The word-based LM was obtained by interpolation of two LMs, an LM trained on 3 hours of transcripts, and second LM trained on pre-processed web-data LM. The final LM was pruned with a threshold 1e-8. The BBN sub-word LM was pruned with threshold 1e-7. For the pruning we used SRILM toolkit.

The lattices I. and III. were converted to HTK format and delivered to BBN for processing in the BBN keyword spotting engine.

Kaldi Keyword spotting

Two KWS engines were used for keyword search:

A. Kaldi search in pre-computed FST indices [Can et al. 2011] (used with lattices I.)
B. Latt2Multigram search in phone confusion networks [Szöke(2010)] (used with lattices I.,II.,III.)

The FST-based engine "A." is efficient for in-vocabulary keywords. For out-of-vocabulary keywords we need engine "B.", in this case the lattices are converted into phone-confusion networks by Minimum Bayes Risk decoding, where a scale of 0.7 is used to both the acoustic and graph scores. Then an exact match search in the phone confusion network is performed, using a phone sequence of the searched term. During the search, we prune-out tokens with sum of log-posterior scores below -25.0, or alternatively the tokens get pruned if a temporal gap between 2 phones is larger than 0.3s.

Kaldi KWS system contains a basic hit-list score calibration techniques: KST (keyword-specific threshold) for FST engine ‚A', and STO (sum to one) for confusion-network based engine ‚B'. For this reason the OracleTWV is more interesting measure than MTWV, if we assume that the more powerful BBN calibration is almost optimal.

Language	Lattices, KWS engine	MTWVnorm [%] (basic calibration)	**Oracle (IV/OOV)[%]**
Cebuano	**I.A. :** Word,FST	20.4	**34.7(40.8/0.0)**
	I.B. : Word,CN	14.7	**32.0(31.7/34.0)**
	II.B. : Phone,CN	11.3	**27.9(27.1/32.9)**
	III.B. : Subword,CN	13.3	**30.0(29.5/32.6)**
Kazakh	**I.A. :** Word,FST	29.1	**44.2(51.0/0.0)**
	I.B. : Word,CN	17.0	**34.3(35.4/27.1)**
	II.B. : Phone,CN	11.9	**28.0(28.1/27.4)**
	III.B. : Subword,CN	15.4	**32.1(32.6/29.2)**
Kurdish	**I.A. :** Word,FST	10.8	**23.9(27.6/0.0)**
	I.B. : Word,CN	6.0	**21.3(22.3/15.1)**
	II.B. : Phone,CN	5.1	**17.8(18.2/15.1)**
	III.B. : Subword,CN	6.2	**20.7(21.0/18.2)**
Lithuanian	**I.A. :** Word,FST	37.0	**48.8(57.2/0.0)**
	I.B. : Word,CN	28.3	**44.4(44.4/43.9)**

	II.B. : Phone,CN	19.2	**34.3(33.7/37.5)**
	III.B. : Subword,CN	26.7	**42.2(41.3/47.3)**
Telugu	**I.A. :** Word,FST	14.1	**26.7(32.8/0.0)**
	I.B. : Word,CN	10.1	**25.3(25.9/22.7)**
	II.B. : Phone,CN	6.5	**18.0(17.8/18.9)**
	III.B. : Subword,CN	9.4	**24.2(24.1/24.8)**
TokPisin	**I.A. :** Word,FST	32.4	**47.7(51.4/0.0)**
	I.B. : Word,CN	19.8	**36.5(37.3/25.8)**
	II.B. : Phone,CN	12.6	**29.3(29.9/21.3)**
	III.B. : Subword,CN	18.1	**34.4(35.0/27.0)**

Table 21: KWS results for tune-set of dev-languages, VLLP, using dev+tune+eval keywords

From table 21, it is clear that the best Oracle-TWV for In-Vocabulary keywords (IV) is with the combination 'I.A.' which corresponds to FST engine applied to word lattices. The best combination for the Out-Of-Vocabulary (OOV) keywords is 'III.B.', i.e. the sub-word lattices converted into confusion networks with phones, where we match the phonetic pronunciation of a keyword. On the other hand the other systems can still be complementary, which is why the 'raw' hit-lists from all the 4 systems were delivered to BBN for further processing.

Babel-related Scholarly Activities including papers published and presentations given during this period

Presented 2 papers at Interspeech 2014:

Martin Karafiat, František Grézl, Karel Vesely, Mirko Hannemann, Jan Cernocky: „BUT 2014 Babel System: Analysis of adaptation in NN based systems“

Frantisek Grezl, Martin Karafiat: „Combination of Multilingual and Semi-Supervised Training for Under-Resourced Languages

Presented 2 papers at SLT 2014:

Martin Karafiat, Karel Vesely, Igor Szoke, Lukas Burget, František Grezl, Mirko Hannemann, Jan Cernocky: „BUT ASR System for Babel Surprise Evaluation 2014“

Frantisek Grezl, Ekaterina Egorova, Martin Karafiat: „Further Investigation into Multilingual Training and Adaptation of Stacked Bottle-Neck Neural Network Structure“

References

- [Hirsch 2005] H.Hirsch and H.Finster, ``The simulation of realistic acoustic input scenarios for speech recognition systems," in Proceedings of Interspeech 2005, Lisabon, Portugal, 2005.
- [Grezl et al.(2009)] F. Grezl, M. Karafiat, and L. Burget, “Investigation into bottle-neck features for meeting speech recognition.” in Proc. Interspeech 2009, no. 9, 2009, pp. 2947–2950.
- [Hinton et al.(2012)] G. Hinton, L. Deng, D. Yu, G. Dahl, A. rahman 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, November 2012.
- [Mohri et al.(2008)] M. Mohri, F. C. N. Pereira, and M. Riley, “Speech recognition with weighted finite-state transducers.” in Handbook on Speech Processing and Speech Communication, Part E: Speech recognition, L. Rabiner and F. Juang, Eds. Heidelberg, Germany: Springer-Verlag, 2008, p. 31.
- [Novotney et al.(2009)] S. Novotney, R. Schwartz, and J. Ma, “Unsupervised acoustic and language model training with small amounts of labelled data.” in Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, ser. ICASSP ’09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 4297–4300. [Online]. Available: http://dx.doi.org/10.1109/ICASSP.2009.4960579

- [Povey et al.(2011)] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The Kaldi speech recognition toolkit.” in Proc. ASRU. IEEE, 2011.
- [Povey et al.(2012)] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiat, S. Kombrink, P. Motlicek, Y. Quian, N. Thang Vu, K. Riedhammer, and K. Vesely, “Generating exact lattices in the WFST framework.” in Proc. ICASSP. IEEE, 2012, pp. 4213–4216.
- [Swietojanski et al.(2013)] P. Swietojanski, A. Ghoshal, and S. Renals, “Revisiting hybrid and gmm-hmm system combination techniques.” in Proc. IEEE ICASSP 2013, 2013.
- [Szöke(2010)] I. Szöke, “Hybrid word-subword spoken term detection.” Ph.D. dissertation, 2010.
- [Szöke et al.(2008)] I. Szöke, L. Burget, J. Cernocky , and M. Fapso, “Sub-word modeling of out of vocabulary words in spoken term detection,” in Proc. 2008 IEEE Workshop on Spoken Language Technology. IEEE Signal Processing Society, 2008, p. 4.
- [Talkin(1995)] D. Talkin, A robust algorithm for pitch tracking (RAPT). New York: Elsevier, 1995, pp. 495–518.
- [Vesely(2011a)] K.Vesely, M.Karafiat, and F.Grezl, Convolutive bottleneck network features for LVCSR, in Proceedings of ASRU 2011, 2011, pp. 42--47.
- [Vesely(2012)] K.Vesely, M.~Karafiat, F.Grezl, M.Janda, and E.Egorova, The language-independent bottleneck features, in Proceedings of IEEE 2012 Workshop on Spoken Language Technology. IEEE Signal Processing Society, 2012, pp.336--341.
- [Vesely(2011b)] K.Vesely, M.Karafiat, and F.Grezl, Convolutive bottleneck network features for LVCSR, in Proceedings of ASRU 2011. IEEE Signal Processing Society, 2011, pp. 42—47.
- [Vesely et al. 2013a] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, “Sequence-discriminative training of deep neural networks.” in Interspeech, 2013.
- [Vesely et al. 2013b] K. Veselý, M. Hannemann and L. Burget: “Semi-supervised Training of Deep Neural Networks”, in Proceedings of ASRU 2013, Olomouc, CZ, pp 267—272.
- [Grezl 2014a] F. Grezl, M.Karafiat and K.Vesely, ``Adaptation of multilingual Stacked Bottle-Neck neural network structure for new language," in Proceedings of Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE, Florence, Italy, May 2014.
- [Grezl 2014b] F.Grezl and M.Karafiat , ``Adapting multilingual neural network hierarchy to a new language," in Proceedings of The 4th International Workshop on Spoken Language Technologies for Under-resourced Languages (SLTU'14)}, St. Petersburg, Russia, May 2014.
- [Grezl 2014c] F.Grezl, E.Egorova and M.Karafiat, "Further investigation into multilingual training and adaptation of stacked bottle-neck neural network structure," in Proceedings of 2014 Spoken Language Technology Workshop. IEEE Signal Processing Society, 2014, pp. 48--53.

- [karafiat 2012] M. Karafiat, M. Janda, Jan Cernocky and L. Burget, ``Region dependent linear transforms in multilingual speech recognition," in Proceedings of International Conference on Acoustics, Speech, and Signal Processing 2012. 2012, pp. 4885--4888, IEEE Signal Processing Society.
- [Ghahremani 2013] P. Ghahremani, B. BabaAli, D. Povey, K. Riedhammer, J. Trmal, S. Khudanpur: “A pitch extraction algorithm tuned for automatic speech recognition” in Proceedings of ICASSP 2014, Florence, Italy, May 2014
- [Laskowski 2008] K. Laskowski, M. Heldner, J. Edlund : “The fundamental frequency variation spectrum” in FONETIK 2008.
- [Zhang et al.(2006)] B. Zhang, S. Matsoukas, and R. Schwartz, “Recent progress on the discriminative region dependent transform for speech feature extraction”, in Proc. of Interspeech 2006, Pittsburgh, PA, USA, Sep 2006, pp. 2977–2980.
- [Can et al. 2011] D. Can and M. Saraclar, “Lattice indexing for spoken term detection,” IEEE Transactions on Audio, Speech & Language Processing, vol. 19, no. 8, pp. 2338–2347, 2011.

GLOSSARY

ASR Automatic Speech Recognition

CER, SER, WER: Character-, Syllable-, Word Error Rate

CN: Confusion Network

DBN: Deep Belief Network

DNN: Deep Neural Network

FLP: Full Language Pack

GMM: Gaussian Mixture Model

KWS: Keyword Spotting

LLP: Limited Language Pack

LM: Language Model

MLP: Multi-Layer Perceptron

ML: Maximum Likelihood

MMIE: Maximum Mutual Information Estimation

MTWV: Maximum Term-Weighted Value

OOV: Out-Of-Vocabulary

PLP: Perceptual Linear Predictive features

RBM: Restricted Boltzmann Machine

RDT: Region Dependent Transform

SAT: Speaker Adaptive Training

SGD: Stochastic Gradient Descent

SST: Semi-Supervised Training

STT: Speech-To-Text

VAD: Voice Activity Detection

VLLP: Very Limited Language Pack

VTLP: Vocal Tract Length Perturbations