Babelon

Period –End Technical Report

Period of Performance: 5 June 2013 – 4 June 2014 (Base Period)

Organizations:	BBN Technologies (BBN) Brno University of Technology (BUT) Johns Hopkins University (JHU) Vocapia Research (LIMSI) Massachusetts Institute of Technologies (MIT) North-West University (NWU)	
Principal Investigators:	Dr. John Makhoul Tel: 617-873-3332 Fax: 617-873-2473 Email: makhoul@bbn.com	Dr. Stavros Tsakalidis Tel: 617-873-4976 Fax: 617-873-2473 Email: stavros@bbn.com
Reporting Period:	5 June 2013 – 4 June 2014	

Babel systems at BUT

Similarly to last year period, 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. The details on system development and insights are described in the following sections.

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 modelling 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 latest NN tricks such as convolutive-bottleneck networks with shared weights [Vesely(2011b)] and RBM pre-training by Contrastive Divergence algorithm.

We use the STK systems mainly to build discriminative features based on Neural Networks and Region Dependent Transforms – often referred to as "BUT features".

These features are based on the concatenation of three feature streams:

PLP HLDA (39 dimensional).

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 decorrelation.

F0 with delta and acceleration coefficient (3 dimensions)

This results in a 72-dimensional feature stream which is further processed by Region Dependent Transforms (RDT) [Zhang et al.(2006)] reducing the dimensionality also from 72 to 69. 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 and 12 Gaussian mixtures per state. They were trained from scratch using mixing-up maximum likelihood (ML) training. Plain SBN systems (no SAT, RDLT) were built for a big part of the analysis, they are referred later as STK BN systems. The final word transcriptions are decoded

using a 3-gram language model (LM) trained only on the transcriptions of the training data. First, the HMM models were trained on PLP features and after that were retrained by single-pass retraining to the BUT69 features. Next, 12 ML iterations followed to better fit the HMMs into the new feature space. The HMM models trained on PLP features were also used to prepare the initial NN training targets by forced-alignment to the transcripts. The alignments are later re-generated by the full system to re-train the full SBN structure.

This year, the SBN architecture was extended by adding a 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.

Kaldi systems

Kaldi [Povey et al.(2011)] is a quickly emerging 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.

With Kaldi, we initially build a GMM-HMM system using flat-start mixing-up ML training on a subset with shorter sentences. The following steps are performed on the full training dataset, we add cross-word triphone tied-states (step ”tri3b”), LDA+MLLT transform on a splice of 9 speech frames (step ”tri4b”), and speaker dependent fMLLR transform (step ”tri5b”). For both FullLP and LimitedLP, we use approximately 4500 tied-states, where each FullLP state is modelled by 24 Gaussians, in LimitedLP by 6 Gaussians. Both the LDA+MLLT features and fMLLR features are 40 dimensional.

This fMLLR system is later used to define tied-states, produce DNN targets by forced-alignment, and provide SAT features for DNN training. The SAT features can eventually be replaced by BUT69 features which led to significant performance improvement. Later analysis has shown that this was probably an effect of a cross-toolkit combination.

The DNN training recipe features generative RBM pre-training, frame classification training, sequence-discriminative training, and semi-supervised training. A more detailed description of the training procedure can be found in chapter 0.

Stacked Bottle-Neck feature extraction

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 1. We originally used 15 Critical-Band Energies (CRBE), with conversation-side-based mean 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.

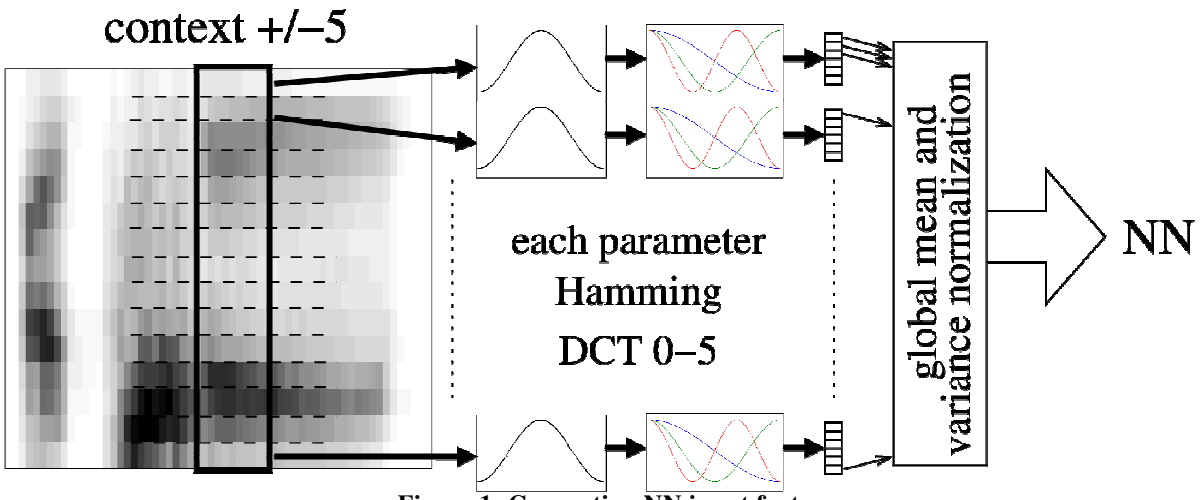

Figure 1: Generating NN input features.

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.

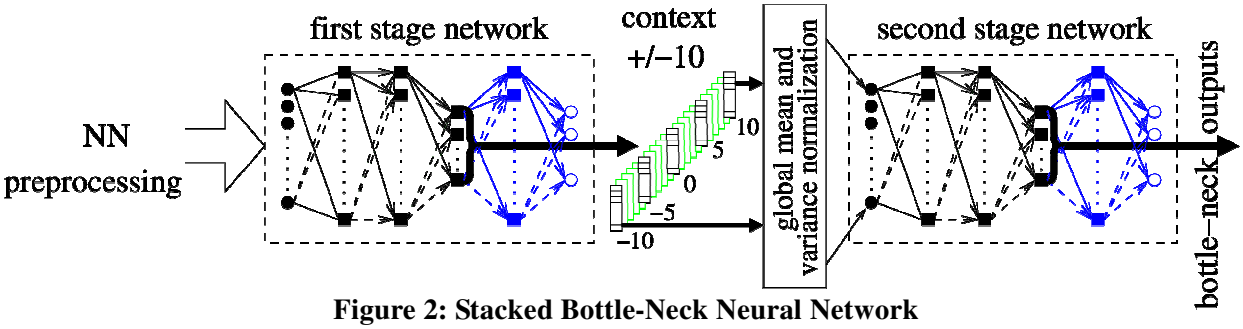

Figure 2: Stacked Bottle-Neck Neural Network

More informative SBN input

One of the first experiments tested richness of the NN input. As mentioned before, we originally used 15 Mel-Filter banks energies for 8kHz speech and phoneme-state labels, as it was found

sufficient in pre-BABEL NN experiments. Currently, we are using significantly bigger NN, therefore a NN can yield from detailed input information.

Input NN features	Vietnamese WER[%]
15FBANK + butF0 + pVoicing	53.8
24FBANK + butF0 + pVoicing	53.3

Table 1: Effect of using more wide filter-banks in NN training, STK BN system.

Results on Vietnamese FullLP show that using 24 FBANK gives 0.5% improvement over 15 FBANK baseline. We also experimented with bigger values and obtained no further improvement. Therefore, we switched to 24 FBANKS in later SBN configurations.

Further improvements in semi-supervised SBN training

In year 1 BABEL period, we investigated bootstrapping approach of Semi-supervised training (SST) of the SBN architecture. The dependence of NN on the quality of 1-best transcripts was examined. Using all utterances with confidence higher than 0.5 was found as a safe value for the NN training (it covered about 70% of un-transcribed data).

The strong gain due to semi-supervised training was over 3% abs., it seems that quality of BN features is relatively insensitive to accuracy of training targets, as the partially incorrectly transcribed sentences were still helpful in training the BN network.

The BN feature extraction is less sensitive to transcription errors but still too many errors can have negative influence on the current performance. Therefore, we experimented with ``fine-tuning'' of NN trained on Supervised + Unsupervised data by re-training on the Supervised data. The learning rate for the ``fine-tuning'' was set to one tenth of its original value. Only the second stage NN is tuned to keep the training process simple and fast. Table 2. shows additional ~0.5% absolute improvement obtained by this procedure.

System	Assamese WER[%]	Bengali WER[%]
NN LLP	71.9	72.2

NN LLP + unsup	67.6 (-4.3%)	69.0 (-3.2%)
NN LLP + unsup -> adapt to LLP	66.7 (-0.9%)	68.4 (-0.6%)

Table 2: Effect of fine-tuning semi-supervised NN with transcribed data, STK BN system.

Cleaning the transcripts for fine-tuning

On Zulu, we also experimented with cleaning the LLP transcripts, which are used to "fine-tune" the semi-supervised LLP system. The data was selected according to the highest allowed Oracle Error Rate (OER) of the lattices generated by ASR of the training data.

OER[%]	All (Inf)	10	20	30	70
Data size [h]	10.1	7.1	8.8	9.5	9.9
WER[%]	69.3	69.2	68.9	69.1	69.0

Table 3: Effect of cleaning supervised transcripts during the fine-tuning from SST, STK BN system.

The table 3 shows 0.4% additional improvement reached by excluding 1.3h of the most problematic data (with the highest OER).

Phone-state targets in SBN

Next, we experimented with using NN targets based on triphone states instead of phoneme states as it is common in DNN-HMM based systems. The triphone-state posteriors are estimated by DNN. The results are on Assamese and Bengali (LLP) with using the latest style of SST:

System	Assamese WER[%]	Bengali WER[%]
NN LLP+unsup -> adapt to LLP (phnstate targets)	66.7	68.4

NN LLP+unsup -> adapt to LLP (xrwdstate targets)	65.5 (-1.2%)	68.2 (-0.2%)

Table 4: Using triphone states as targets in NN based systems, STK BN system.

Moving from phoneme states to triphone states NN targets gives 1.2% gain on Assamese and 0.2% gain on Bengali.

Various F0 input features in SBN architecture.

Fundamental frequency (F0) is an important feature in speech recognition systems of tonal languages. Estimating the pitch in the signal is sensitive to errors, therefore pitch trackers are using many techniques like dynamic programming to increase robustness. Precision of F0 systems can also vary across the processed phonemes, therefore we experimented with using more ``complementary'' F0 estimators as an additional features to NN classifier.

We experimented with 4 fundamental frequency (F0) estimators. The first three are based on normalized cross-correlation function.

- BUT F0 - Our tool which was implemented according to [Talkin 1995].
- GetF0 - tool using snack library (http://www.speech.kth.se/snack).
- Kaldi F0 - F0 estimation recently implemented in Kaldi [Ghahremani, et al, 1994].
- Fundamental Frequency Variations (FFV) provides continuous vector-valued representation of F0 variation. It is obtained by comparing the harmonic structure of the frequency magnitude spectra of the left and right half of an analysis frame [Laskowski, et al, 2008].

An SBN NN was trained on various F0 estimators to investigate the impact of pitch on tonal and non-tonal languages.

Additive NN input	Bengali WER (%)	Haiti WER (%)	Lao WER (%)
No F0	70.6	67.0	65.5
BUT F0	70.9	66.8	64.3
(B)BUT F0+pVoicing	70.6	66.7	64.2
GetF0	70.5	66.9	64.4
KaldiF0	70.1	66.5	63.3
KaldiF0+pVoicing	69.7	65.7	62.8
(K) KaldiF0+pVoicing+Delta	**69.5**	**65.6**	**62.6**
FFV	70.0	66.6	64.2
(B)+(K)	69.7	65.5	62.3
(B)+(K)+FFV	69.2	65.4	62.3
(B)+(K)+FFV+GetF0	69.3	65.4	62.2

Table 5: Effect of using a various F0 in NN training, STK BN system.

Table 5. presents dominant effect of Kaldi F0. The NN reads a context of +/- 15 frames, therefore it should learn information about variation of F0, interestingly delta F0 is still giving 0.2% improvement, probably due to computation of this feature in unnormalized space [Ghahremani, et al, 1994], therefore it still contain complementary information.

Fusion of all F0 estimators gives 0.1-0.4% absolute improvement over Kaldi F0. Moreover, we can clearly see that F0 features are well effective also for non-tonal languages.

Reiteration of unsupervised transcriptions in SST of SBN system

Significant improvements coming from SST trained motivated us do one more decoding of unsupervised data and retraining of NN, to see a dependency of system performance on a quality of unsupervised transcripts.

System	Bengali WER (%)	Haiti WER (%)	Lao WER (%)
LLP	69.5	65.4	61.7
SST-1	66.2	61.1	56.9
SST-2	65.6	59.5	55.6
SST-2 → LLP	65.1	59.1	55.0

Table 6: Regeneration of unsupervised transcripts, STK BN system.

Table 6 presents 3.3-4.6% absolute improvement coming from NN trained on SST data (SST-1). For simplicity, the system was evaluated on SBN features only using ML GMM models and LLP data. The improvements were encouraging, therefore we experimented with regeneration of unsupervised data by system based on SST-1 NN. The new NN SST-2 was trained which gave further ~0.5% absolute improvement.

Speaker adaptation of SBN architecture

The speaker adaptation of NN is a problem that has not been entirely solved. A typical speaker adaptation uses standard cepstral coefficients (PLP or MFCC) together with CMLLR as NN input; on the other hand, Mel-filter bank outputs (FBANK) are known to work better with NN but their direct adaptation is difficult due to correlations. Our trick to adapt the FBANKS is very simple:

To estimate the adaptation matrix, it is necessary to train a GMM system on the NN input features, however, FBANKs are difficult to model by diagonal covariance models due to high correlations. This problem is solved by using Discrete Cosine Transform (DCT) in the same way as in MFCC computation. Next, the speaker independent GMM HMM system is estimated by single-pass retraining with FBANK-DCT appended by derivatives and acceleration coefficients. A block-diagonal CMLLR transform is estimated for each speaker and only the first, spectrum corresponding, part of the transform is taken for further processing. New features for NN training are estimated simply by:

$$\hat{x}(t) = x(t) A_{DCT} A_{CMLLR} A_{invDCT}$$

Next, we focused on adaptation of the SBN inner part - the first stage NN. A bottle-neck output is known not to be highly correlated, therefore the CMLLR can be applied easily. Moreover, according to our analysis, the first-stage NN is doing mainly acoustic feature extraction and only the second-stage NN is doing processing acoustic clues in wider context. Therefore, it makes

sense to use speaker-specific layer in this part as it is common in classical speech recognition scenarios (feature extraction, speaker adaptation, acoustic modeling).

NN adapt level	Vietnamese WER (%)
No NN adapt	52.3
CMLLR on FBANK	51.1
CMLLR on 1stageNN+MLLT	48.9
CMLLR on 1stageNN	48.9

Table 7: Effect of using NN adaptation, STK BN system.

Multilingual SBN systems

The availability of sufficient amount of data from other BABEL languages motivated us to investigate how to use it to improve the performance in the target language. In the first year of the Babel program, we already experimented with the training of Multilingual SBN.

Three main approaches were examined:

1. "one softmax" - discriminates between all targets of all languages. No mapping or clustering of phonemes was done. Thus the resulting NN has quite a large output layer containing all phonemes from all languages with "one softmax" activation function.

2. "block softmax" - divides the output layer into parts according to individual languages. During the training, only the part of the output layer corresponding to the language the given target belongs to, is activated. This approach was successfully used in [Vesely et al.(2012)].

3. "Convolutive Bottleneck Network" which would allow to re-train the whole structure in one step. But the computation expenses to train the multilingual NN and also to adapt this network were unacceptable for our scenario where fast adaptation is desired.

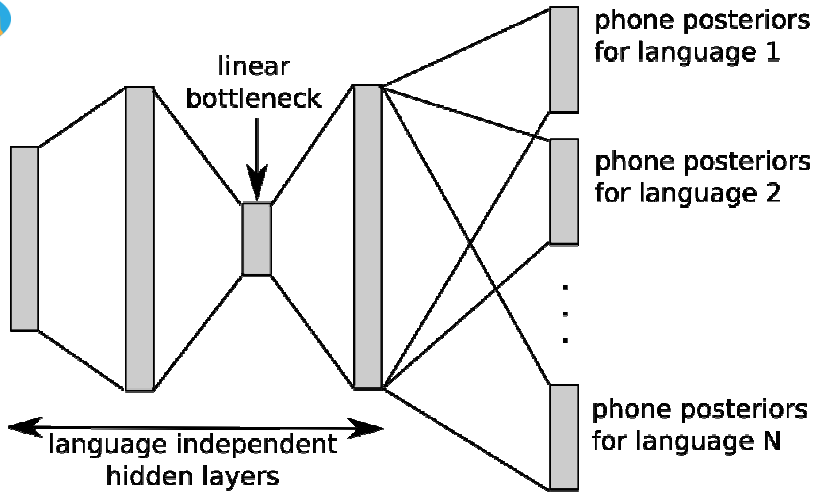

Figure 3: Block softmax multilingual NN.

This multilingual NN provided the starting point for the final "fine-tuning" to target language. It was produced by further NN training but the learning rate in this phase was set to one tenth of its original value.

The combination of "block-softmax" and "Convolutive BN" depending on the language was found to be the best approach. This year, we further investigated in optimal "fine-tuning" configuration as well as improvements coming from more data and also on implementation of Multilingual NN together with CMLLR adaptation introduced above.

The Multilingual NN was trained without Kaldi F0 as this feature was found at the end of Babel period. The results are analyzed on HMM-GMM system trained on BN features only due to simplicity.

Multilingual SBN system – amounts of data

First set of experiment was focused on the performance of BN features obtained from multilingual NNs. The NNs were trained on FLPs of the source languages. To be able to evaluate the effect of the number of source languages, we decided to generate two sets of them:

1. SLs1 - contains three language: Cantonese, Pashto and Turkish.
2. SLs2 - contains all five languages from year one.

Data sets	Assamese WER (%)	Bengali WER (%)	Haiti WER (%)	Lao WER (%)	Zulu WER (%)
FLP	61.5	62.9	57.2	55.1	68.9
LLP	68.5	69.7	65.9	63.6	74.2
SLs1	69.0	69.6	66.8	64.7	74.1
SLs2	66.8	68.2	64.9	60.7	72.6

Table 8: Effect of using more data for Multilingual NN training, STK BN system.

The Table 8 presents significant improvement (over 1.4%) coming from using more languages in multilingual NN training without any "fine-tuning". So, the number of training languages plays an important role when the features are generated directly from the multilingual system. The "FLP" line present LLP system trained on top of FLP system which is treated as an "upper-bound". On the other-side "LLP" shows results of systems with NN trained purely on LLP transcripts. Therefore, SLs2 NN presents better performance even no target language was presented in NN training.

Multilingual SBN system – tuning of the configuration

The SBN system is a hierarchy of two NNs, so the adaptation can take several forms. We can for example keep the first NN multilingual and train the second one on the LLP data only. Thus we considered the following four scenarios for adaptation of multilingual system:

- Multi-LLP scenario : keep the first NN multilingual, train the second one on LLP data only.
- Adapt-LLP scenario : adapt the first NN, train the second one on the LLP data.
- Multi-adapt scenario : keep the first NN multilingual, adapt the second one.
- Adapt-adapt scenario : adapt the first and also the second NN.

The last "adapt-adapt" scenario might be regarded as not very good idea since the inputs to the second NN are changed but, surprisingly, out initial experiments discovered that this scenario is not useless.

Since every multilingual SBN system was adapted according to all four scenarios for each of the evaluation languages, showing all results would not provide an easy survey. To be able to present the results together, each one is converted to relative WER reduction with respect to the corresponding LLP baseline, and averaged over the evaluation languages.

For better readability, the results are split into two plots based on the Source Language set used to train the multilingual SBN systems.

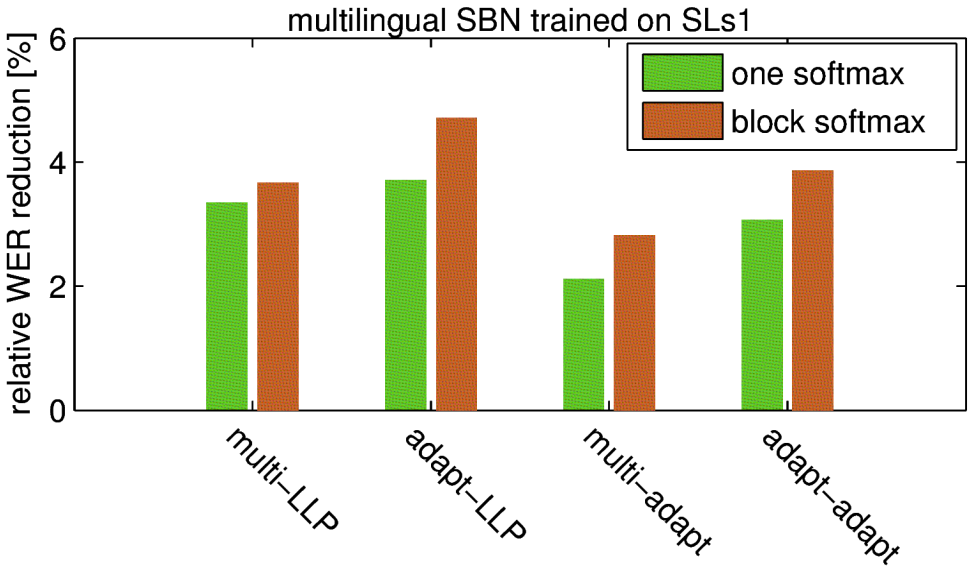

Figure 4: Average relative WER [%] reduction depending on the adaptation scheme and softmax type in multilingual SBN on 3langs.

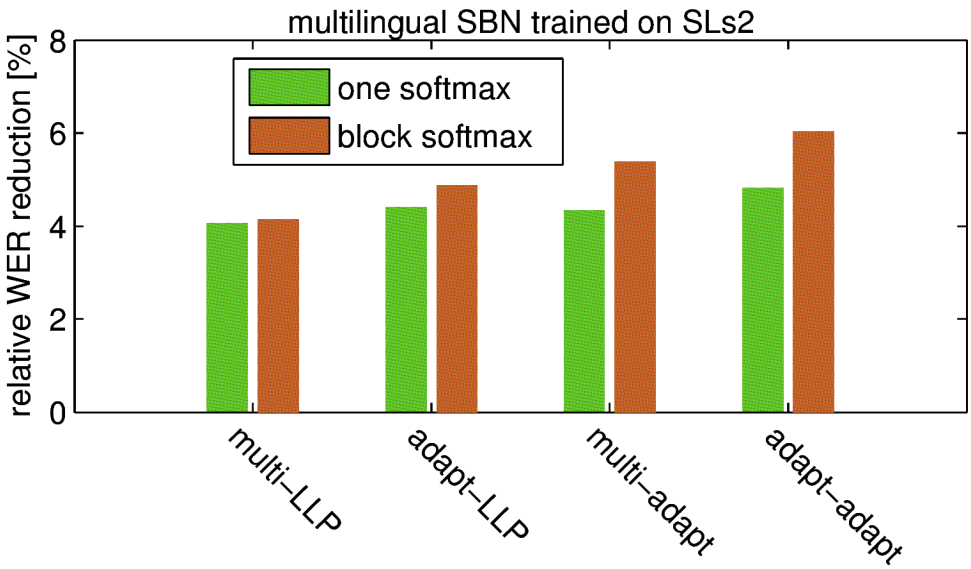

Figure 5: Average relative WER [%] reduction depending on the adaptation scheme and softmax type in multilingual SBN on 5langs.

Figs. 5 and 6 show that “multi-LLP” scenario leads to the same performance regardless the softmax type in multilingual SBN. This suggests that this scenario cannot take advantage from the differently trained SBN. When the first NN is adapted before the training of the second one on LLP in “adapt-LLP” scenario, the WER can be further reduced. Note the increased difference between different types of softmaxes in multilingual SBN - the adaptation process can benefit more from the SBNs trained with “block softmax”.

In case of training on SLs1, the WER is reduced the least when only the second NN is adapted in “multi-adapt” scenario. The last scenario “adapt-adapt” when both NNs in SBN are adapted, actually performs about the same as “multi-LLP” scenario. This shows that also the second NNs in the hierarchy can be adapted despite its changed inputs by former adaptation of the first NN.

The picture changes when the multilingual NNs are trained on SLs2. The adaptation of the second NN only in ”multi-adapt” scenario outperforms the “adapt-LLP” one and adapting both NNs “adapt-adapt” - leads to systems with the best performance.

Multilingual SBN - fine-tuning to CMLLR adapted NN features

Our final system is based on SBN architecture with CMLLR adaptation of 1stageNN as it was presented before. Therefore, multilingual 1stageNN was "fine-tuned" into target LLP domain and fixed. Next, the CMLLR features were generated for whole LP including unsupervised data. These features were used for further “fine-tuning”/training of 2stageNN (Merger). Three different scenarios were investigated:

1. Merger was retrained directly on CMLLR features (mult-LLP).
2. Fine-tuning of multilingual merger into CMLLR features (mult-adapt).
3. Fine-tuning of multilingual merger which was already tuned on non-adapted LLP features (adapt-adapted).

NN system	WER [%]				
	Assamese	Bengali	Haiti	Zulu	Lao
LLP					
Multi-LLP	62.3	-	58.6	69.0	56.1
Multi-adapt	62.6	-	58.9	69.9	55.6
Adapt-adapted	62.3	-	58.7	69.9	55.6
SST					
Multi-LLP	61.2	-	57.3	68.7	54.4
Multi-adapt	61.6	-	58.0	69.1	54.4
Adapt-adapted	61.5	-	57.8	69.1	54.5

Table 9: Multilingual NN - "fine-tuning" to 1stage CMLLR domain, STK BN systems.

2StageNN can be retrained on LLP (or SST) data from random initialization which is opposite outcome from no-CMLLR SBN.

Final BUT GMM features

Final ASR system structure did not change during this period, it still consist from PLP-HLDA features concatenated with SBN features and F0 followed by RDT. Main effort was put into SBN structure improvement which is the fundamental part of GMM system and BUT features.

Most of the particular improvements in SBN architecture led into rebuilding of the whole system. These feature versions were used across the team:

- BUTv2 – SST, CMLLR NN, 24FBANKS, 3xF0 (no KaldiF0)
- BUTv2a + SST->LLP (fine-tuning from SST NN into clean LLP domain)
- BUTv3 + + reiteration of unsupervised data + Kaldi F0

System	WER [%]		Normalized MTWV [%] All (IV/OOV)
	Assamese	Zulu	Zulu
BUTv2a	60.4	69.0	-
BUTv3	59.0	68.5	18.74 (35.31/0)
Multiling-noSST	59.6	68.6	-
Multiling-SST	58.8	68.4	18.85 (35.51/0)

Table 10: Final STK systems in 2014 evaluations.

Good improvement from multilingual systems over BUTv2a but small improvement over BUTv3. It was mainly due to Kaldi F0, but retraining of whole multilingual NN is quite time consuming operation.

Deep Neural Network Kaldi systems

In the last 4 years, Deep Neural Networks (DNN) became very popular for acoustic modeling. The DNNs are used to compute posterior probabilities of triphone tied-states, which are used as emission probabilities in the HMM decoder. This is often called as a ``hybrid'' system.

A big advantage of DNNs over GMMs is that DNNs are trained discriminatively to classify frames, whereas GMMs are typically trained generatively. Also, in case of DNNs all the states share the same hidden layers, which leads to efficient use of parameters and diminishes the risk of under-fitting the rare tied-states. Another advantage is that, they DNNs don't require strong assumptions on input features. One can simply splice several speech frames and feed to DNN input, which will produce highly correlated features. All these advantages translate into huge performance improvements, when comparing a DNN and GMM system trained on the same input features.

On the other hand, to be fair, we have to say that the performance of a GMM can be greatly improved when using bottleneck features (called ``tandem systems''), which make them equally as good as DNNs, while still giving complementary outputs.

To decode the DNN output, we need to convert posteriors into log-likelihoods, this is done according to Bayes rule by subtracting the state log-priors. For the decoding, we can use the pre-softmax activations, as the decoder input is not required to be normalized. This is also the case when we decode using GMM likelihoods.

In the first year of Babel program we applied these training techniques:

- Generative RBM pre-training, using Contrastive divergence algorithm with 1 Gibbs sampling step.
- Mini-batch Stochastic Gradient Descent optimizing frame-level cross-entropy. Here, we first train using GMM alignments and then repeat the training using DNN alignments. The triphone tree is inherited from the baseline GMM system.
- Sequence discriminative training optimizing 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 hypothesis represented as a lattice, while the reference path is obtained by forced-alignment.
- Semi-supervised training – with frame-rejection based on a per-frame confidences derived from lattice posteriors selected according to the best path [Vesely et al.(2013)b].

Important changes compared to year 1 systems

In the year 2 of BABEL program, we have been modifying the DNN systems, while looking for performance improvements. The interesting and useful techniques were:

- Adding more states to the system
- Simplification of semi-supervised training
- Input features:
    - Replacing the BUT pitch feature by Kaldi pitch features
    - Replacing the Kaldi PLP-fMLLR features with Stacked bottleneck features

Adding tied-states

In year 1 LimitedLP systems, we were using 2300 tied-states, however we believed that this can limit the space for improvement coming from semi-supervised training. As we have only 10h of transcribed data, we decided to keep the number of Gaussians constant while increasing the number of tied-states to 4300 (same value as for FullLP systems). The results show a 0.1% performance degradation for GMM systems, and 0.2% improvement of DNNs.

Bengali WER (%)	2300 tied-states	4300 tied-states
fMLLR GMM	69.7	69.8
DNN Xent (GMM alignment)	64.7	64.4
DNN Xent (DNN alignment)	64.3	63.9
+ sMBR	62.2	62.4
+ SST (Xent only)	61.8	61.6

Table 11: Effect of adding more tied-states on semi-supervised DNN system.

Simplification of semi-supervised DNN training

In our previous work [Vesely at all (2013b)], we converged into a recipe where the semi-supervised training is used for frame Cross-Entropy training, while for the sMBR training, the transcribed 10 hour dataset is used. In the semi-supervised training, we ended up using all the automatically transcribed sentences, repeating the transcribed data 3x, and rejecting 18% of frames according to a per-frame confidence threshold tuned to 0.7. We also found that frame-

weighted training is as good as un-weighted training. Note, that we implement frame-weighted training by scaling the gradients in error back-propagation.

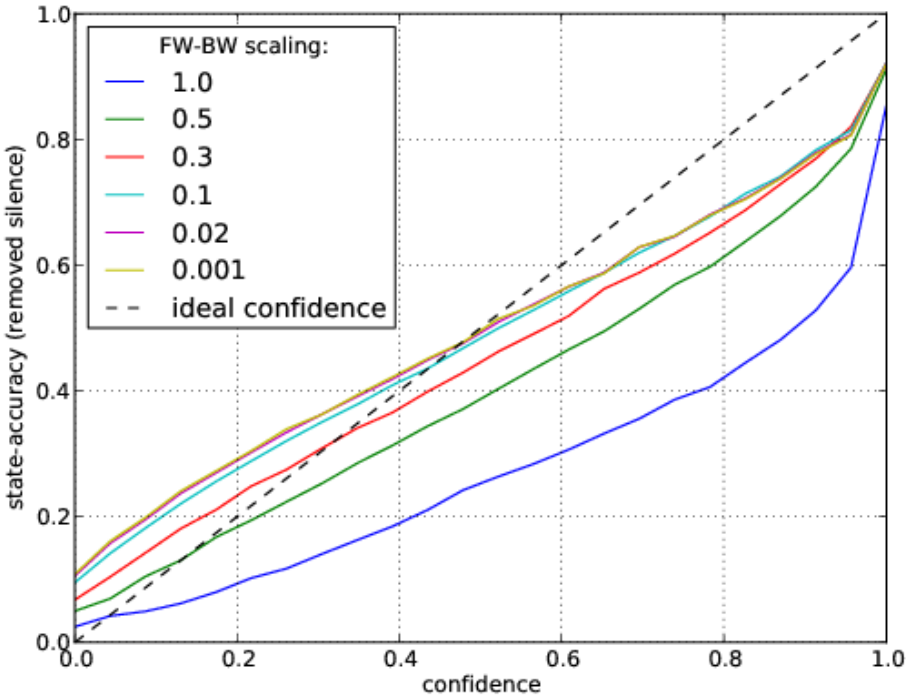

Fig 6: Dependency of real state accuracy on estimated confidence.

In this BABEL period, we continued analyzing of the confidences. In picture 9, we can see the state-accuracy as function of the confidence for different lattice-posterior scaling factors, where we calculated the fraction of correct frames for 25 bins on the confidence axis. The dashed black line is the "ideal confidence" corresponding to the probability that label is correct. The blue line corresponds to the original scale 1.0, which corresponds to the setup used in decoding. This picture suggest, that we were previously using an overly ``optimistic" confidence, which was giving a higher confidences compare to the actual state-accuracies. When we scaled down the lattice posteriors, we obtained nearly linear dependency between the state-accuracy and the confidence, which should be true for the ideal confidence.

The effect of the lattice-posterior scales on the confidence-weighted training is in table 10. We see that for scale 0.02 we get 0.3% lower WER, compared to original scale 1.0.

Scaling	1.0	0.5	0.3	0.1	0.02	0.001
WER (Bengali)	61.3	61.1	61.2	61.1	61.0	61.1

Table 12: Tuning the lattice-posterior scale during per-frame confidence extraction for frame-weighted training.

Compared to the previous recipe, we found out that it is no longer necessary to tune frame-rejection threshold, neither to include the supervised data in multiple copies. This is what makes the new semi-supervised training more efficient and simpler.

Kaldi pitch features

As the team Radical was promoting Kaldi pitch features, while altruistically making them available to other teams, we compared the pitch feature to BUT pitch. As can be seen in table 11, using Kaldi pitch led to a large 1.7% WER improvement on Bengali which is atonal language:

Bengali WER (%)	BUT pitch	Kaldi pitch
DNN with sMBR training	69.5	67.8

Table 13: Replacing BUT pitch by Kaldi pitch.

The BUT pitch is implemented according to [Talkin 1995], it is normalized by mean pitch over the entire speaker and linearly interpolated across the unvoiced gaps. Along the pitch also the voicing feature is extracted.

The Kaldi pitch [Ghahremani 2013] is composed of 3 features: the pitch, delta-pitch and voicing feature. The pitch 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.

Input features from Kaldi

Our GMM feature set are PLPs (13 dimensions) concatenated with fundamental frequency (Kaldi F0) and probability of voicing. 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):

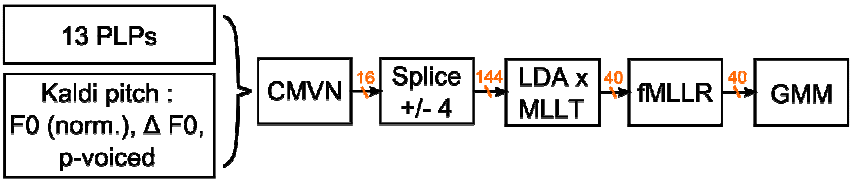

Figure 7: Feature extraction pipeline for Kaldi GMM systems

The fMLLR features are then used as DNN input, here we splice 11 frames (440 dimensions), and apply shift and scale so the features have zero mean and unit variance.

Figure 8: Input features for Kaldi DNN/DBN systems

Using BN features as DNN inputs

As mentioned above, we used the STK features as front-end to Kaldi DNN. These 80-dimensional features were extracted from the first stage bottleneck and were rotated by speaker specific fMLLR transform.

These features were concatenated to the standard Kaldi fMLLR front-end, while the bottleneck features were processed as in the Stacked-bottleneck network: we concatenated 5 frames with temporal offsets of -10, -5, 0, 5, 10 frames w.r.t. the central frame. Using these input features, the DNN systems were then built using the same procedure as before, including sequence Minimum Bayes Risk (sMBR) and semisupervised training.

System	WER [%]					
	Bengali		Haiti		Lao	
STK GMM:						
BUTv3	60.5		53.8		50.7	
Kaldi DNN, with front-end:						
Stage:	Kaldi	Kaldi+STK	Kaldi	Kaldi+STK	Kaldi	Kaldi+STK
Basic	64.4	60.3	59.4	54.0	55.0	50.7
+ sMBR	62.0	58.5	57.4	52.9	53.2	49.9
+ + SST	59.8	56.8	54.4	50.8	50.8	48.3

Table 14: Using 1Stage BN features in DNN system, ASR results.

The Table 14 presents the very impressive results. Over 4% absolute improvement is reached in final SST system by adding STK features and more than 2.5% absolute improvement over the best GMM-SST trained system!! Features used as a DNN input are SST trained therefore both techniques are well complementary.

KWS on Limited LP with using BBN calibration:

System	MTWV [%]				
	Assamese	Bengali	Haiti	Zulu	Lao
GMM					
BUTv3	31.35	32.02	45.38	20.38	45.94
DNN (PLP CMLLR)					
sMBR + SST	34.00	33.89	42.54	21.45	47.24
DNN (PLP CMLLR + BN CMLLR)					
sMBR + SST	-	37.08	45.47	-	50.20

Table 15: Using 1Stage BN features in DNN system, KWS results.

- Significant gain was observed with new DNN based on adapted BN on all languages. We are getting slight gain only on Haitian, which should lead into complementary systems.

Noising the data

In Surprise evaluation we had to deal with various noises in the data. Artificial noises were added into training data to increase a robustness of the system. 10 types of noises were ad-hoc generated mostly (v1-v8) by filtering of white-noise:

- **v1 - v4 -** low pass filters - where {\bf v1} is close the one we have in data.
- **v5, v6** - one band-pass filter.
- **v7, v8** - several band-pass filters in cascade.
- **v9** - 100Hz hum (sinus).
- **v10** - 50Hz hum (sinus).

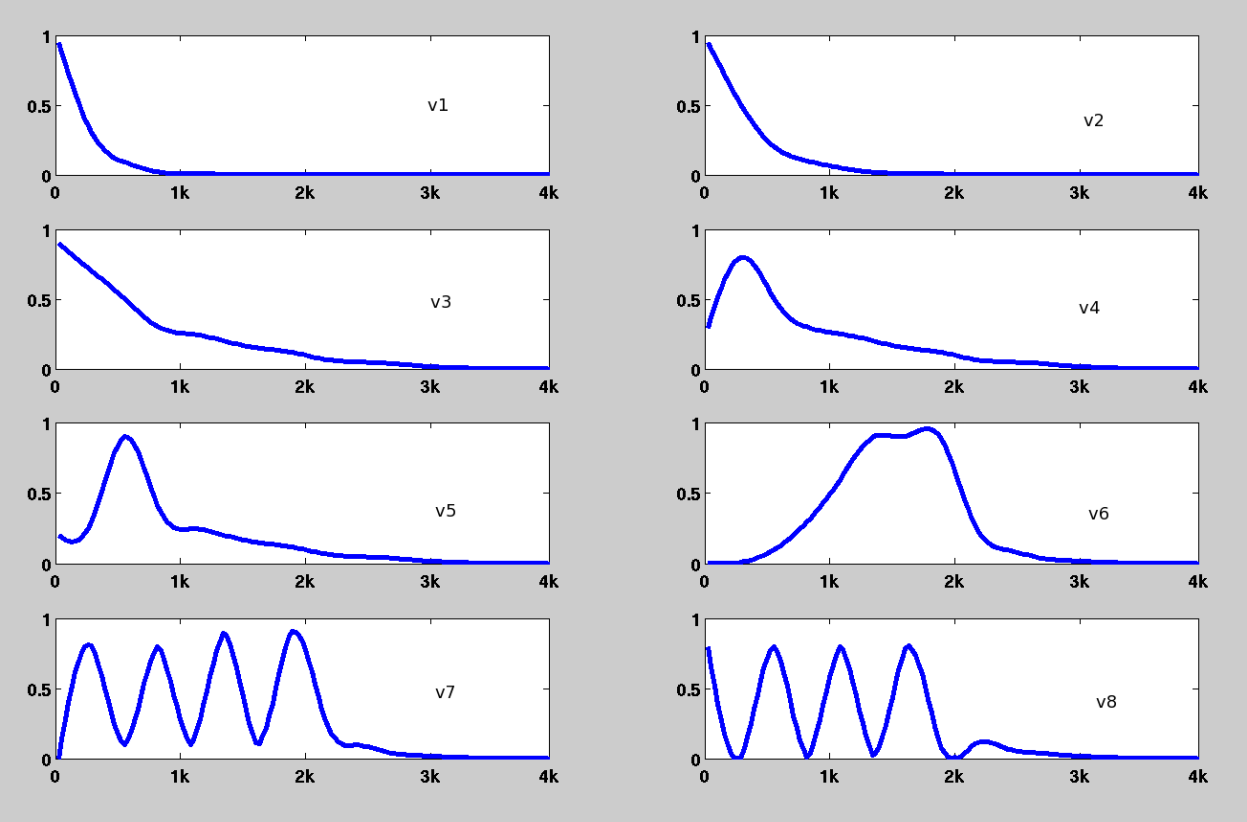

Figure 9: Frequency characteristics of noise v1 – v8, v1 - v4 low pass filter, v5 – v6 band pass, v7 – v8 multiple band pass.

Noises were added to the data - according to RMS of the data. We use three noising levels:

- L1 – 0.35 of RMS of the original signal.
- L2 – 1.00 of RMS of the original signal.
- L3 – 3.5 of RMS of the original signal.

Therefore, we multiplied our training data by factor 4 (original data + 3 levels of noises). All features (PLP, FBANK, F0) were regenerated and NN training target was multiplied from clean data. The NN was trained on the new data and the RDT and GMM system was rebuild on clean data only which create system called ``BUTv4" and also on all data - ``BUTv4a".

ML trained BN systems	Tamil WER[%]
Baseline	76.0
Wiener filter on train/test	76.1

Adding additive noise in training data	75.4

Table 16: Wiener denoising vs. data noising on NN system, STK system.

Final GMM results:

GMM tandem systems	WER [%]	KST normalized MTVW All (IV/OOV) [%]
Baseline	72.1	19.36 (29.93/0.00)
Added noises to NN only	71.7	19.73 (30.49/0.00)
Added noises also to RDT-GMM	71.2	20.32 (31.40/0.00)

Table 17: Adding noises to NN only and also to RDT, STK system.

Data-noising significantly increases robustness of the system.

Using noised data for NN training is complementary even it we train GMM on the clean data only.

SST trained features from 1stageNN trained on noised data were taken as an input into DNN (results are with sMBR+SST training).

HMM-DNN hybrid systems	WER[%]
Baseline	69.4
1stageBN on noised data DNN on clean only	68.1
DNN is trained also on noises	67.7

Table 18: Feeding 1StageNN CMLLR trained on noises into DNN, STK system.

For the DNN system, artificial noising of the training data is also helpful, both on input bottleneck-feature level and the DNN level.

System	Normalized MTWV (IV/OOV) [%]				
	Assamese	Bengali	Haiti	Zulu	Lao
Word Lattice	28.6 (36.7/0)	29.3 (38.4/0)	43.2 (50.4/0)	18.8 (35.4/0)	42.8 (47.2/0)
Word CN	31.8 (40.8/0)	31.2 (40.9/0)	43.5 (50.7/0)	20.9 (39.3/0)	42.9 (47.3/0)

Table 19: Comparison between lattice and confusion networks for word based systems of dev languages.

Word Lattice – baseline keyword spotting technique based on word lattice search. This system suffers from missing OOV words (term having OOV is not detected – 0.00 TWV for OOV terms) and also can suffer from multiple word keywords (in case there is no path in lattice representing multiple word in sequence, these keywords are not detected – see improvement when going from WRD to WRD CN)

Word CN – keyword spotting system based on word confusion networks. Standard lattice can be converted into confusion network. The improvement going from CN is in better recall over multiword keywords. This can be seen in comparison between WRD and WRD CN.

System	KST normalized MTWV (IV/OOV)	Oracle TWV (IV/OOV)
WRD	18.46 (28.53/0.00)	27.17 (41.99/0.00)
WRD CN	20.32 (31.40/0.00)	29.32 (45.31/0.00)
MGRAMWBWE	23.86 (25.80/20.72)	35.41 (40.55/26.00)
MGRAMPHN2PHN CN	23.38 (20.83/28.71)	38.63 (37.64/40.44)
BBNSYL2PHN CN	23.76 (20.49/29.91)	39.67 (38.26/42.27)

Table 20: Comparison for several KWS methods on surprise language.

- WRD – baseline Word Lattice keyword spotting technique
- CN – denotes usage of confusion networks
- MGRAMWBWE – Keyword spotting based on word internal phoneme multigrams (length up to 7 phonemes). The keyword is searched as phoneme multigram sequence in multigram lattice.
- *2PHN CN – We implemented new approach for subword search. Here, the STK decoder (Svite). The decoder was switched to produce model (triphone) nodes in lattice rather than the word nodes. This is independent on the used language model (can be word, multigram or phoneme). Then the triphones in lattices were converted into phonemes and lattices were stored in (HTK format). Next the phoneme lattice was converted to CN and search was performed. Here we found crucial to set up 2 parameters. The first one was pruning, where the search is stop if accumulated posterior of a keyword drops below the threshold (to speed up the algorithm). Next we set up a maximum length of skip links in search keyword in CN. Without this, the detection can reach theoretically to the length of the whole utterance (and is not correct). We found optimal value of the maximum length as 0.1s.

We can conclude that:

- MGRAMWBWE is superiors to WRD systems due to its possibility to find OOV terms (29.32 → 35.41).
- When going from multigram search in lattice (MGRAMWBWE) to phoneme search in Confusion Network (*2PHN CN) we lose some precision on IV terms (compared to word internal multigrams 40.55 → 38.26), but we gain much on OOV terms (26.00 → 42.27). So the overall UBTWV improves twice much the improvement when going from words to MGRAMWBWE.
- The KST normalization is not sufficient enough, because we can see improvement on UBTWV (the Oracle TWV) but not on MTWV. This is solved by more complex normalization in BBN.

BUT Decoding and Keyword spotting – further analysis

Baseline systems

System	MTWV (IV/OOV)	Oracle TWV (IV/OOV)	# detections
WRD – 3g LM	28.16 (39.13/0.00)	37.90 (52.67/0.00)	319k
WRD CN – 3g LM	31.15 (43.29/0.00)	39.83 (55.34/0.00)	480k
MGRAMWBWE – 3g LM	31.19 (36.83/16.87)	41.81 (49.68/21.61)	209k
MGRAM – 3g LM	31.05 (36.14/18.16)	43.59 (50.26/26.46)	311k
MGRAM – 2g LM	31.07 (35.74/19.13)	44.40 (50.50/28.75)	368k

Table 21: comparison of baseline systems on Assamese LLP

According to the table above, we clearly see that confusion networks are superior to lattices (WRD versus WRD CN). It is also represented by higher number of detections. We compare subword search based on phoneme multigrams (up to 7 phonemes) on last 3 rows. The MGRAMWBWE represents word internal multigrams compared to MGRAM which are word external multigrams. The word internal multigrams provides higher accuracy on IV keywords while word external multigrams provides higher accuracy on OOV keywords. The OOV accuracy can be even higher by going from trigram to bigram LM.

One can argue, that the number are not comparable due to different number of detections, but our experiments shown, that when we set the systems to produce comparable number of detections (by decoder pruning), the conclusion is the same.

New subword KWS based on phoneme CNs

System	MTWV (IV/OOV)	Oracle TWV (IV/OOV)	# detections
WRD2PHN CN – 3g LM	21.90 (26.31/10.84)	32.06 (38.10/16.56)	587k
MGRAMWBWE2PHN CN – 3g LM	23.08 (25.52/16.90)	35.50 (39.75/24.60)	746k
MGRAM2PHN CN – 3g LM	23.92 (26.46/17.59)	36.37 (40.62/25.45)	1072k

MGRAM2PHN CN – 2g LM	24.25 (26.90/17.63)	36.63 (40.86/25.75)	1209k
MGRAM2PHN CN – 3g LM less pruned	30.17 (31.07/28.24)	46.43 (48.80/40.34)	5283k

Table 22: Comparison of new developed KWS techniques based on phoneme CNs on Assamese LLP

Please refer the *2PHN CN to above section.

We conclude, that to produce phoneme lattice from word LM is suboptimal as you can see in the second row (WRD2PHN CN). Also the word internal multigrams (MGRAMWBWE) do not achieve the accuracy of cross word multigrams (MGRAM). However going from trigram to bigram does not improve the accuracy (opposed to the baseline experiments). Finally, we present selected cross word multigram system when the search pruning was relaxed (the last row), where huge improvement can be seen.

System combination

The above experiments are single system (one decoding with single LM and one search). Now we aimed at the result, when system combination will be used. BBN run the Learner normalization for "MGRAM2PHN CN – 3g LM less pruned" system and combined it with our best word based system (WRD CN)

System	KST MTWV (IV/OOV)	Learner TWV (IV/OOV)
WRD CN	28.61 (36.72/0.00)	33.15 (42.53/0.57)
MGRAM2PHN CN	29.35 (30.03/27.06)	34.44 (35.92/30.27)
COMBINATION of the 2 above	**-**	**41.39 (44.59/30.20)**
BUT-Kaldi-DNN-wrd-lattice:	-	39.68 (44.25/23.78)
BUT-MLP APN sub (sup)	-	30.72 (31.66/27.48)
36 systems combination	-	47.70 (50.47/38.10)

Table 23: comparison of our new KWS approach on Assamese LLP dev+eval terms and dev data (systems submitted to evaluations).

Next, we compared the combination to the best single system for IVs (BUT-Kaldi-DNN-wrd-lattice), best single system for OOVs (BUT-MLP APN sub (sup)), and submitted combination of 36 systems. As we can see the combination of our 2 systems is superior to the best single system on OOVs and comparable to the best single IV system. It looses about 6 point to the 36 system combination.

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

Papers presented at Interspeech 2013 (Lyon, France):

- K. Vesely, A. Ghoshal, L. Burget, D. Povey: Sequence-discriminative training of deep neural networks
- M. Karafiat, F. Grezl, M. Hannemann, K. Vesely, J. Cernocky: "BUT BABEL system for spontaneous Cantonese"

Paper presented at ASRU 2013 (Olomouc, Czech Republic)

- F. Grezl, M. Karafiat and K. Vesely: "Adaptation of Neural Network Feature Extractor for New Language"
- Vesely K., Hannemann M., Burget L.: "Semi-supervised training of deep neural networks"

Papers presented ICASSP 2014:

- Karafiat M., Grezl F., Hannemann M., Cernocky J. : BUT Neural Network features for spontaneous Vietnamese in BABEL
- Grezl F., Karafiat M., Vesely K.: "Adaptation of Multilingual Stacked Bottle-Neck Neural Network Structure for New Language"

Papers presented at SLTU 2014:

- Grezl F, Karafiat M: „Adapting Multlingual Neural Network hiearchy to a new language"

References

- [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.
- [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.

GLOSARRY

ASR Automatic Speech Recognition

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

CN: Confusion Network

DBN: Deep Belief Network

DNN: Deep Neural Network

GMM: Gaussian Mixture Model

KWS: Keyword Spotting

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