Using Gradient Descent Optimization for Acoustics Training from Heterogeneous Data

Martin Karafiát[*], Igor Szöke, and Jan Černocký

Brno University of Technology, Faculty of Information Technology
Department of Computer Graphics and Multimedia, Speech@FIT
Božetěchova 2, Brno, Czech Republic
{karafiat,szoke,cernocky}@fit.vutbr.cz

Abstract. In this paper, we study the use of heterogeneous data for training of acoustic models. In initial experiments, a significant drop of accuracy has been observed on in-domain test set if the data was added without any regularization. A solution is proposed by getting control over the training data by optimization of the weights of different data-sets. The final models shows good performance on all various tests linked to various speaking styles. Furthermore, we used this approach to increase the performance over just the main test set. We obtained 0.3% absolute improvement on basic system and 0.4% on HLDA system although the size of the heterogeneous data set was quite small.

1 Introduction

The amount of in-domain training data has significant effect on the accuracy of speech-to-text transcription systems based on Hidden Markov Models (HMM). Speaking style is often the major cause of variability in the data, therefore only in-domain data are typically used for HMM training. Our target domain is recognition of spontaneous Czech continuous telephone speech (CTS), but good performance on non-spontaneous data (radio) is also desired. In our initial experiments, we observed significant drop off accuracy if models were trained just on CTS and tested on non-spontaneous data.

An intuitive solution is to add non-spontaneous speech to the training process. This is however not trivial as it caused significant drop of accuracy on in-domain (CTS) test set. Weighting of training data is necessary, but doing this manually is difficult especially if the amount of different training data sources is high. Therefore, we defined an objective function, and perform HMM training with respect to minimization of this function. Weighting of training data set could be done by removing some speech segments or better, by weighting of statistics needed for HMM estimation.

1.1 Weighting of HMM Statistics

Forward-backward algorithm is common algorithm used for estimation of HMMs. It generates occupation probabilities $\gamma_j(t)$ for Gaussian mixture component j, which allow to gather sufficient statistics for re-estimation formulae:

[*] This work was partly supported by Czech Ministry of Interior project No. VD20072010B16, Grant Agency of Czech Republic project No. 102/08/0707, Czech Ministry of Education project No. MSM0021630528, and by BUT FIT grant No. FIT-10-S-2.

$$\gamma_j = \sum_{t=1}^{T} \gamma_j(t) \tag{1}$$

$$\boldsymbol{\theta}_j(\mathbf{O}) = \sum_{t=1}^{T} \gamma_j(t)\mathbf{o}) \tag{2}$$

$$\boldsymbol{\theta}_j(\mathbf{O}^2) = \sum_{t=1}^{T} \gamma_j(t)(\mathbf{o}(t))^2 \tag{3}$$

Then, Gaussian parameters are re-estimated according to:

$$\hat{\boldsymbol{\mu}}_j = \frac{\boldsymbol{\theta}_j(\mathbf{O})}{\gamma_j} \tag{4}$$

$$\hat{\boldsymbol{\sigma}}_j = \frac{\boldsymbol{\theta}_j(\mathbf{O}^2)}{\gamma_j} - \boldsymbol{\mu}_j^2 \tag{5}$$

In case of training from various databases **D**, it is possible to divide the collection of statistics into database-specific parts:

$$\gamma_j = \sum_{d=1}^{D} w(d)\gamma_j^d, \tag{6}$$

$$\boldsymbol{\theta}_j(\mathbf{O}) = \sum_{d=1}^{D} w(d)\boldsymbol{\theta}_j^d(\mathbf{O}), \tag{7}$$

$$\boldsymbol{\theta}_j(\mathbf{O}^2) = \sum_{d=1}^{D} w(d)\boldsymbol{\theta}_j^d(\mathbf{O}^2), \tag{8}$$

where $w(d)$ is weight of database d. When starting, all values $d(w)$ are initialized to one. Next, it is possibly to enhance system performance by optimization of objective function $F(M(\mathbf{w}))$, which includes accuracies on different test-sets. It depends on the current model set $M(\mathbf{w})$, where $\mathbf{w}$ is the vector of all weights $w(d)$.

2 System Optimization

2.1 Gradient Descent Approach

Gradient Descent (GD) optimization is a general algorithm that can be directly used for our purpose. It uses derivatives of the objective function to find the steepest gradient. In the process, the optimized variables are moved in negative direction which will reduce the value of the function.

The process is iterative and can be described as follows:

1. Compute the derivative $dF(x)/dx$ of the function $F(x)$ with respect to its independent variables x.

Table 1. Language model training data

Corpora	amount of words [W]
CTS training data	685k
PMKBMK	1182k
subtitles	192M

2. Change the value of x according to

$$x^{(n+1)} = x^{(n)} - \eta \frac{dF(x^{(n)})}{dx}, \tag{9}$$

   where η is learning rate.
3. Repeat above steps till convergence is reached.

The learning rate η is a control value which significantly influences the convergence of the algorithm. If it is too big, a step will overshoot the minimum of $F(x)$. If too small, the process will need long time to converge. Therefore, we used variable learning rate:

$$\eta^{(n)} = \begin{cases} 1.2\eta^{(n-1)} \text{ if } & F(x) \text{ increases} \\ 0.5\eta^{(n-1)} \text{ if } & F(x) \text{ decreases,} \end{cases} \tag{10}$$

which guarantees fast convergence while not overshooting the minimum.

3 Experimental Setup

The speech recognition system is based on HMM cross-word tied-states triphones. MF-PLP features were generated using HTK, with a total number of 13 coefficients. Deltas, double-deltas and (in the HLDA system), triple-deltas were added, so that the feature vector had 39 or 52 dimensions respectively. Cepstral mean and variance normalization was applied with the mean and variance vectors estimated on speaker basis. VTLN warping factors were applied by adjusting the centers of the Mel-filters. HLDA was estimated with Gaussian components as classes and the dimensionality was reduced to 39.

All tests in this paper used 2-gram language models trained on corpora described in Table 1. PMKBMK is Prague and Brno corpus of spoken Czech[1]. The corpus of subtitles was obtained from the web. The size of dictionary was 1M words[2].

3.1 Data Description

The training and test data was collected from various sources which significantly differ in channel and speaking style:

[1] `http://korpus.cz/english/pmk.php`, `http://korpus.cz/english/bmk.php`

[2] Note, that Czech is highly inflective language and 50k dictionary usual in English systems is far too small.

Table 2. Training and test data

Training set	amount of data [h]	Test set	amount of data [h]
train-C	46	testCTS	2.2
train-R	19	testR	1
train-L	21	testL	0.5
train-P	15	testP	0.5
train-V	2.2	-	

Table 3. Effect of adding databases into the training

Models	testCTS	testR	testP	testL
train-C	52.6	54.2	39.5	37.2
train-CR	52.3	63.2	49.3	46.5
train-CP	52.2	58.1	62.4	53.6
train-CL	51.7	59.2	57.3	56.7
train-CV	52.6	56.3	41.4	39.8
train-CRPLV	51.3	64.2	63.0	59.1

- CTS - “train-C” - Spontaneous telephone speech.
- RadioCTS - “train-R” - People calling to radio during broadcasts. Partly spontaneous speech.
- Liberec - “train-L” - Read speech (over the telephone) from University of Liberec.
- Plzen - “train-P” - Read speech (over the telephone) from University of South Bohemia.
- 158 - “train-V” - Emergency calls. Very short spontaneous recordings.

The amounts of data taken for training and test can be found in Table 2.

4 Experiments

As spontaneous telephone speech was our main domain, the initial model set was trained on CTS data only. Next, we investigated the effect of adding each particular database into the training process. The initial models were re-trained by additional iterations of standard maximum likelihood (ML) training. The results are given in Table 3.

We observed that adding any data into the training does not improve and even mostly degrades the performance on “testCTS”. On contrary, initial CTS models have quite poor performance on other tests – a system based on these models will work poorly in case it has to deal with read data instead of spontaneous (which can easily happen in a real application). Using all corpora, without any weighting, significantly improves the performance on all non-CTS tests but causes 1.3% drop on “testCTS” – this is not satisfactory as CTS is our main domain.

To verify, that this misbehavior was caused by different speaking styles, we run same experiment with speaker-based adaptation for testing. This adaptation performs more

Table 4. Effect of adding databases into the training with application of CMLLR

Models	testCTS	testR	testP	testL
train-C	54.6	62.2	46.0	47.6
train-CR	54.6	66.9	56.1	55.3
train-CP	54.1	65.0	64.6	59.3
train-CL	54.3	65.4	61.4	61.7
train-CV	54.7	63.0	48.0	48.2
train-CRPLV	53.8	67.4	65.2	63.6

efficient speaker and channel compensation than basic CMN/CVN used in experiments above. Constrained Maximum Likelihood Linear Regression (CMLLR) was taken for this purpose [1,2]. The results can be found in Table 4. We observed generally better results than in the experiment without adaptation, but the same trends with different training data.

4.1 Gradient Descent Training

It is obvious that system degradation on "testCTS" caused by adding non-CTS data can be reduced by minimizing their influence. It could be simply done by setting up the weighting coefficients to values smaller than one, see equations (6)–(8).

Optimal weights could also be found by exhaustively running many tests, or automatically by optimization of objective function. We defined the objective function $F(\mathbf{w})$ as a weighted sum of word error rates (WER) of the system for each test set. The weights were set according to performances expected on particular test sets.

For our application, we wanted to minimize the accuracy drop on "testCTS" and put smaller importance on other tests, so that the objective function was defined as:

$$\begin{aligned} F(\mathbf{w}) = {} & 0.6A_{testCTS}(M(\mathbf{w})) + 0.2A_{testR}(M(\mathbf{w})) + \\ & +0.1A_{testP}(M(\mathbf{w})) + 0.1A_{testL}(M(\mathbf{w})), \end{aligned} \tag{11}$$

where $A_x(M(\mathbf{w}))$ is WER of the resulting model M on test set x, and $\mathbf{w}$ is vector of weights used for merging of statistics in equations (6)–(8). Obviously, the coefficients multiplying accuracies could be set differently, depending on the target application.

Training acoustic models with respect to optimization of $F(\mathbf{w})$ can be described in the following steps:

1. Initialize weights. The most simple initialization is:

   $$\mathbf{w} = \left[\ 1.0\ 1.0\ 1.0\ 1.0\ 1.0\ \right],$$

   where columns represent weights for training databases
   ["C" "R" "P" "L" "V"]. Train initial models and evaluate $F(\mathbf{w})$.
2. Collect statistics for all training sets.
3. Estimate $dF(\mathbf{w})/d\mathbf{w}$ where $d\mathbf{w}$ is approximated by a small step. This is quite painful operation due to the need to run decoding across all test sets for each

   $$\left[\frac{dF(\mathbf{w})}{dw_1} \cdots \frac{dF(\mathbf{w})}{dw_D}\right]$$

Table 5. Gradient Descent optimization with and without CMLLR

Optimization tests run without adaptation				
Models	testCTS	testR	testP	testL
train-C	52.6	54.2	39.5	37.2
train-CRPLV	51.3	64.2	63.0	59.1
GD optim train-CRPLV	52.1	64.1	61.7	58.3
$\mathbf{w}$ = [1.21 0.92 0.60 0.66 1.60]				
Optimization tests run with CMLLR				
train-C	54.6	62.2	46.0	47.6
train-CRPLV	53.8	67.4	65.2	63.6
GD optim train-CRPLV	54.1	67.0	64.8	62.7
$\mathbf{w}$ = [1.23 0.92 0.95 0.80 1.09]				

4. Update new weights according to equation (9).
5. Normalize weights to sum to the same number as initial weights.
6. Estimate new models set, evaluate $F(\mathbf{w})$ and go back to step 3. These iterations are run until the process stabilizes.
7. Go back to step 2 and iterate until the whole process stabilizes.

Note, that returning to step 2 can change the optimal weights. Therefore, we fix the number of gradient descent iterations to a fixed value, and stop the training when the whole process, including ML iterations, is stabilized.

The final results are shown in Table 5. Degradation of accuracy is now 0.5% instead of 1.3% absolute and the models perform well also for other data.

Using optimized weights in HLDA systems. The Heteroscedastic Linear Discriminant Analysis (HLDA) [3] can be used to derive a linear projection de-correlating feature vectors and reducing their dimensionality. It is nowadays a common technique used for enhancing speech recognition systems. For implementation into this scheme, we investigated two possibilities, both used fixed weights from the optimization of the basic system:

– HLDA is estimated with the initial set of models – training using all data is done by additional ML iterations with fixed GD weights. It is denoted “GD train-C HLDA + RPLV” in Table 6.
– Models coming from the optimization above are used to estimate HLDA statistics from all the data. Statistics are merged with fixed weights, new HLDA transformation matrix is estimated and new models are trained. This is denoted “GD train-CRPLV HLDA” in Table 6.

Table 6 shows smaller drop of accuracy on “test-C” with fixed HLDA than with the retrained HLDA. This is caused by using just close-domain data for HLDA estimation.

Optimization for testCTS only. In initial experiments, we have found that non-spontaneous data has no positive effect in training of CTS system if the weights are set

Table 6. Using GD weights in HLDA system

Models	testCTS	testR	testP	testL
train-C	52.6	54.2	39.5	37.2
train-C HLDA	53.5	56.7	41.8	42.9
GD train-C HLDA + RPLV	53.2	65.9	63.8	60.4
GD train-CRPLV HLDA	52.7	66.2	64.2	61.2

Table 7. Gradient descent optimization for enhancing accuracy just on "testCTS" set

Models	testCTS
train-C CMLLR	54.6
train-C-V CMLLR	54.7
GD optim train-CRPLV CMLLR	54.9
$\mathbf{w} = [\ 3.15 \quad 0.32 \quad 0.14 \quad 0.03 \quad 1.36\ ]$	

Table 8. Gradient descent optimization for enhancing accuracy just on "testCTS" set with using HLDA

Models	testCTS
train-C HLDA CMLLR	56.1
GD train-C HLDA + RPLV CMLLR	56.1
GD train-CRPLV HLDA CMLLR	56.5

to be equal. Therefore, we run an experiment where we changed the objective function just to accuracy on "testCTS":

$$F(\mathbf{w}) = 1.0 A_{testCTS}(M(\mathbf{w})) \tag{12}$$

This is similar to an approach often used in language modeling – use out-of-domain corpora for enhancing the performance on current task [4].

The initial weights were set close to the expected final values, only "train-C" and "train-V" weights were set to non-zero values, as they show positive effect on accuracy:

$$\mathbf{w} = \left[\ 3.0\ 0.0\ 0.0\ 0.0\ 2.0\ \right],$$

The whole optimizations run with CMLLR in the tests. Table 7 presents 0.2% absolute gain from models trained on improving data only ("train-CV") and 0.3% against "train-C" baseline. On the resulting weights, we see that main importance was put on in-domain data and partly on "train-V" and "train-R" training sets. The read speech corpora contribute almost zero.

In experiments in Section 4.1, we have shown no positive effect on "testCTS" from retraining HLDA on all data if weights are optimized to produce balanced model for all tests. But if weights are optimized for this test, we can benefit from re-estimation of HLDA transformation on all data – Table 8 shows 0.4% absolute improvement.

5 Conclusions

In this paper, we have studied the use of heterogeneous data for the training of acoustic models. To obtain desired performance, we used regularization based on optimization over the data weights. The regularization was found to be very useful to increase robustness of acoustic models to various speaking styles, or for use of heterogeneous data for a single target application. In case of enhancing the training data, we obtained 0.3% absolute improvement for the basic system and 0.4% for the HLDA one. Note, that the amount of heterogeneous data was smaller than for in-domain data. It is therefore realistic to expect even more improvement with increased size of out of domain data data. In our future work, we will investigate the possibility to run the optimization together with Speaker Adaptive Training [5,2].

References

1. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1), 1–38 (1977)
2. Gales, M.: Maximum Likelihood Linear Transformations for HMM-Based Speech Recognition (1997)
3. Kumar, N.: Investigation of Silicon-Auditory Models and Generalization of Linear Discriminant Analysis for Improved Speech Recognition. Ph.D. thesis, John Hopkins University, Baltimore (1997)
4. Iyer, R., Ostendorf, M., Gish, H.: Using Out-of-Domain Data to Improve In-Domain Language Models. IEEE Signal Processing Letters 4(8), 221–223 (1997)
5. Tsakalidis, S., Byrne, W.: Acoustic Training from Heterogeneous Data Sources: Experiments in Mandarin Conversational Telephone Speech Transcription. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005), March 18-23, vol. 1, pp. 461–464 (2005)