Automatic Speech Segmentation Based on HMM

Martin Kroul

Inst. of Information Technology and Electronics, Technical University of Liberec, Hálkova 6, 461 17 Liberec, Czech Republic

martin.kroul@tul.cz

Abstract. *This contribution deals with the problem of automatic phoneme segmentation using HMMs. Automatization of speech segmentation task is important for applications, where large amount of data is needed to process, so manual segmentation is out of the question. In this paper we focus on automatic segmentation of recordings, which will be used for triphone synthesis unit database creation. For speech synthesis, the speech unit quality is a crucial aspect, so the maximal accuracy in segmentation is needed here. In this work, different kinds of HMMs with various parameters have been trained and their usefulness for automatic segmentation is discussed. At the end of this work, some segmentation accuracy tests of all models are presented.*

Keywords

Speech processing, automatic segmentation, speech database, HMM, monophones, triphones, alignment.

1. Introduction

In today's speech applications, large speech databases are used frequently. For many of them, high-accurate segmentation has to be done. In the past, a manual segmentation has been used mostly. It was hard work, took a lot of time and required an experienced person. Today this is becoming impossible, because databases containing many hours of speech utterances are very common.

Speech synthesis can be a good example. For phoneme synthesis (wide-spread in the 70th and 80th years, but not used any more because of a bad quality of synthesized speech) around 40 speech units is needed. For diphone synthesis (wide-spread in the 90th years) it can be up to 1.600 and for today mostly used triphone synthesis it can be around 30.000 speech units. It is obvious, that speech database for phoneme synthesis usually contains only few sentences and can be segmented manually without problems. Speech database for diphone synthesis can contain several minutes of speech and still can be segmented manually. But for triphone synthesis database creation we need several hours of speech utterances. This amount of data cannot be segmented manually any more, so it is necessary to use some kind of automatic segmentation in this case. Another example of automatic segmentation necessity can be a data preparation for the initialization phase of a HMM training.

2. Automatic Segmentation

Most of today's automatic segmentation methods are based on speech recognition algorithms using DTW (Dynamic Time Warping) [1, 2, 3] or HMM (Hidden Markov Models) [1, 2, 4, 5], but we can also use methods based on speech signal or frequency spectrum change-points, for example SVF (Spectral Variation Functions) [6].

Speech modeling with HMMs is considered as the best method for automatic segmentation today, therefore it will be described here in detail. Three-state models of monophones or triphones are common in continuous speech recognition applications. The number of mixtures is usually 32-64 for monophones and 3-8 for triphones. Automatic segmentation is based on speech recognition, so identical models and parameters can be used for it. For automatic segmentation of an utterance, its model composition is needed first, created by monophone/triphone models concatenation. This composite model is used by Viterbi algorithm for finding the most probable assignment of speech frames and model states. With knowledge of this assignment, frames located on borders of monophone/triphone models (parts of the composite model) can be declared as phoneme borders. The Viterbi algorithm output probability shows, how the model $\boldsymbol{M}$ matches to the speech signal $\boldsymbol{X}$ and can be used for model quality (accuracy) determination. It is defined as:

$$P(X,M) = \max_{w} \sum_{f=1}^{F} t_{w(f)w(f-1)} p_{w(f)}\left(x_f\right) \tag{1}$$

where w is the Viterbi sequence of model states maximizing the $P(\boldsymbol{X},\boldsymbol{M})$, $t_{w(f)w(f-1)}$ is the probability of the transition from the state visited in frame $f-1$ to the state aligned to frame f and $p_{w(f)}(x_f)$ is the probability that the vector x_m is emitted by the latter state. The assignment of speech frames to model states is called the ***forced alignment*** [7]. Illustration of this method can be found in [2].

2.1 Signal Framing

For further processing of an utterance and its recognition, speech signal segmentation into short-time parts called frames is needed. To avoid confusion, this kind of segmentation will be further called framing. Framing is signal division into short parts with the same length. These parts have to be short enough to be stationary and long enough to give us sufficient information at once. For better signal description, these frames are overlapped, as shown in Fig. 1.

For continuous speech recognition, 20-25 ms long frames are used mostly, with the 10 ms frame rate (the frame rate is a time between two incoming frames), so the length of overlap is about a half of the frame length. For automatic segmentation, these values are insufficient. After recognition, in the phase of backward assigning of frames to model states, we can find border frames only, not exact border points in speech samples. Border sample *n* between the two neighboring frames $Frame_1$ and $Frame_2$ can be determined as

$$n = mid(Frame_1) + \frac{mid(Frame_2) - mid(Frame_1)}{2} \quad (2)$$

where $Frame_1$ is the last frame assigned to the first model, $Frame_2$ is the first frame assigned to the second model and *mid()* establishes the position of the middle sample of the frame. So with the 10 ms frame rate, we cannot determine the border point with more than 10 ms accuracy. The higher frame rate we use (less time between two incoming frames), the more accuracy we can achieve. In our experiments the 3ms frame rate has been used.

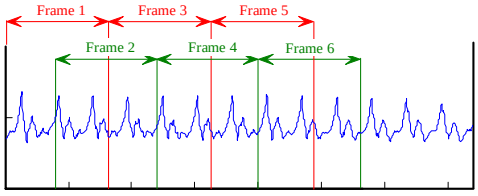

Fig. 1. Speech signal framing.

2.2 Speech Corpus

For successful model training, large amount of training data has to be used. For continuous speech recognition purpose, we usually need to create speaker and environment independent models. So the training database has to contain various recordings from many speakers, male and female, recorded in different conditions. The more various aspects we include the better models (more universal) we obtain. Good sources of this kind of data are radio and television. In this database, we can mix TV and radio news, sport, weather forecast and discussion programs. If we train models only on one-speaker training data, we can use them only for this speaker's utterances recognition. For other speakers, we obtain much worse recognition score. For high-quality independent models training, we need tens of hours of training data.

In this contribution, we focused on automatic segmentation of recordings obtained from one speaker, which will be used for triphone synthesis unit extraction. For this kind of automatic segmentation, the training database will be absolutely different from the one for continuous speech recognition. The goal is to recognize one-speaker utterances, all of them recorded in the same conditions. We don't need speaker-independent universal models, so we don't need many speaker recordings in the training database. Actually it is undesirable, because speaker-independent model recognition is always worse, than speaker-dependent (recognition with models trained on data of the same speaker). On the contrary, speaker independent models are usually more robust, than speaker dependent, because it is always easier to record several utterances from various speakers, than a lot of utterances from only one speaker.

Robustness is a very important indicator in model training. 100 frames is an amount of data, recommended as minimum for confidential determination of one Gaussian function parameters [7]. For quality model training, at least 100 frames per one model mixture are necessary. For some of phoneme models this can be a problem. In phonetic transcription of our Czech training database, containing about 36000 phonemes, the „ó“ phoneme was found only 34 times, what means less than 0,1%. In 50 minutes long speech recording, this phoneme filled less than 8 seconds. With 3 ms frame rate, this represents about 2700 frames. Let's imagine the following. With three-state HMM we cannot expect uniform distribution of frames into states (3 x 900). The first state usually represents the start of the phoneme, the second its middle and the third its end (transition to next phoneme). The second (middle) state uses to be the longest and contains most of frames. With the theoretical distribution 10% frames to 1st state, 80% frames to 2nd state and 10% frames to 3rd state, we can assume for the „ó“ phoneme the following distribution: 1st state - 270 frames, 2nd state - 2160 frames and 3rd state - 270 frames. This amount of training data is sufficient for one-mixture models, where for each state only one Gaussian function is computed. In this case, for more than one-mixture models, 100 frames per mixture rule could be violated already. There is a different assignment of frames to each mixture, so for two mixtures, the distribution could be 70:200 for example. In the training database creation phase, there is very important to care about a sufficient amount of all phonemes, to keep models robust enough and to avoid recognition score decrease.

2.3 Model Training

As mentioned before, for successful recognition and automatic segmentation, quality phoneme models are needed. Model parameters are obtained from statistical analysis of a large amount of training data. This process is called model training and consists of two parts.

For the first phase called ***Initialization,*** a small amount of segmented data is needed (automatically or

manually). For each phoneme, every occurrence of it in these data is found and its initial model parameters (feature means and variances) are computed. If these models consist of more than one state, Viterbi algorithm is used in several iterations to determine the optimal frames-to-states distribution. At the end of this phase we have phoneme models, which could be used for recognition and automatic segmentation already. But the recognition score or the segmentation quality would be low (depends on the amount of the training data and the quality of its segmentation). If no segmented data are available, a method called „Flat Start" [4, 7] can be used. This algorithm computes means and variances from all training data regardless the meaning and use them for each model. So after that, every phoneme model will have the same parameters – something like an „average" phoneme. These models are not usable for recognition, they are only prepared for the next phase. The Flat Start method isn't used very often, because it makes recognition results worse.

The second phase is called ***Reestimation*** and improves the accuracy of model parameters, obtained in the initialization phase. It is based on Baum-Welch algorithm [7], which assigns speech frames to model states like Viterbi algorithm, but doesn't need segmented data on input (needs only phonetic transcription of the utterance). The other difference is that each frame is not assigned to one state only, but belongs with a certain probability to every state of the model. This improves flexibility significantly. In the training phase, phoneme borders are not fixed, so speech frames located on the phoneme borders can be assigned to both models. This enables diffusion of neighboring states.

In this phase, a large amount of speech recordings can be used. The more is the better. Several iterations of Baum-Welch algorithm have to be done to achieve the best result. In each iteration, speech frames are newly redistributed into states and then model parameters are updated. In the following iteration, previous computed models are used. In continuous speech recognition, about ten iterations are made usually. With more iterations, the effect called overtraining can occur. Models are still better focusing on the training data, but with different data recognition (other speaker, microphone) the score gets worse. This case is dangerous for models, which will be used for speaker-independent recognition, but in case of automatic segmentation, where the training data will be recognized, the overtraining could be a valuable option.

In training of our models, the following assumptions were used:

- The better initialized models we use, the better models we get after reestimation (this is the reason why Flat Start shouldn't be used).
- The more training data we use, the better and more robust models we get.
- The more different sources of data we have, the more universal models we get.
- For one-speaker utterances recognition, speaker-dependent models are better than universal.

From these assumptions we decided, that for automatic segmentation, a large amount of one-speaker data should be used for initialization and reestimation. Maximum of Baum-Welch algorithm iterations should be made, until models get better.

For comparing quality of models, the logarithmic probability, obtained from Viterbi and Baum-Welch algorithm in the training phase (equation 1) can be used. It is computed as the cumulated product of frame assignment probabilities with optimal assignment frames to states. In practice, average logarithmic probability per frame is often used. It is always a negative number, usually in range from -70 to -40. The higher the average logarithmic probability is the more accurate the models are.

3. Experiments

For model training, we had about 650 MB of speech data (the total length of recordings was 5 hours and 38 minutes).

Individual parts were labeled as following:

- **Data1**: One-speaker's recordings (a man), which will be used for triphone synthesis unit extraction and hence needs to be automatically segmented.
- **Data1_MS**: Manually segmented part (15%) of Data1.
- **Data1_AS**: Data1, automatically segmented with common continuous speech recognition monophone models, obtained from the Speech Lab at the Technical University of Liberec (64 mixtures, frame rate = 10 ms, frame length = 20 ms).
- **Data2**: Recordings of various speakers. They will be used for models training only.
- **Data2_MS**: Manually segmented part (21%) of Data2.
- **Data2_Male:** Male recordings from Data2.
- **Data2_RS_Male:** Male recordings from Data2_RS.

For speech recordings parameterization, the following options were used:

- frame length = 20 ms,
- frame rate = 3 ms,
- number of features = 39

(13 MFCCs and their first and second derivations)
For parameterization, HTK software [7] has been used.

3.1 Monophones

First of all, three-state monophone models have been trained using the HTK software. These models are context-

independent and their number use to be equal to the number of phonemes. For each monophone, all occurrences of this phoneme in the training data will be used for training. The results of the training are shown in Tab. 1. All models were trained with 12 iterations of Baum-Welch reestimation algorithm. In the first two cases, 8-mixture models were trained, keeping satisfactory robustness even for the least occurred phonemes. 32-mixture models were trained then. Although there were not enough training data for satisfactorily training the least occurred phonemes, in result these models were better, then the 8-mixture models.

Nr.	Mix.	Init.	Reest.	Log. Prob.
1	8	Data1_MS	Data1	-64,042
2	8	Data1_AS	Data1	-63,876
3	32	Data1_MS	Data1	-62,569
4	32	Data1_AS	Data1	-62,158
5	32	Data1_MS+ Data2_MS	Data1	-62,715
6	32	Data1_MS+ Data2_MS	Data1+ Data2	-63,410
7	32	Data1_MS+ Data2_MS_Male	Data1	-62,660
8	32	Data1_MS+ Data2_MS_Male	Data1+ Data2_Male	-63,226

Tab. 1. Monophone models training results.

Other conclusions follow:

- For model initialization, all available data are better to be used, although they are not segmented onto phonemes accurately, than less accurately segmented data.
- The more training data of one speaker are available, the more accurate models will be obtained.
- Speaker-dependent models are better for automatic segmentation than speaker-independent ones.
- The more similar the training data are to the data for segmentation (only men recordings), the better models will be.

Although the models trained on automatically segmented data were best in result, manual correction of some phoneme borders was needed before training. The phoneme models with fewer occurrences in training data couldn't be trained at all, because of insufficient training data (due to wrong automatic segmentation, some phoneme lengths were set to almost zero).

In Fig. 2, the average logarithmic probability per frame is shown after each of iterations. The models number 2 and 4 (initialized on automatically segmented data) are improving only a little, only two or three iterations are sufficient to use. For the models initialized on manually segmented data, more iterations are needed, 10-12 is sufficient.

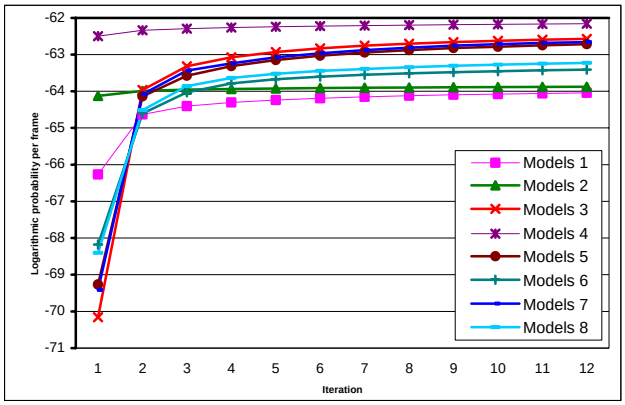

Fig. 2. Average log. probabilities per frame after each iteration.

3.2 Triphones

Three-state triphone models were trained next. Triphones are context-dependent phonemes, so they can model speech signal better, than monophones (especially the coarticulation). In monophones training, all occurrences of each phoneme were used to create model, regardless the neighboring phonemes. In triphone models training, several models are created for each phoneme, regarding its left and right context. The number of models depends on training parameters. Triphone training process is a little bit more complicated, than the monophone one. For triphone models initialization, one-mixture monophone models are needed. Parameters of triphone models, derived from the same monophone model, are simply copied from this monophone. Transition matrices of all triphones, derived from the same monophone, are very similar, so they can be copied and kept unchanged for the whole training process. This brings an advantage of robustness preservation. The number of speech frames, usable to train triphone model, will be much less, than the number of frames usable for monophone model training (frames used for training of one monophone have to be divided to train all the derived triphones). So with keeping transition matrices fixed, unreliable parameter estimation is avoided. Because there will be the same matrices for more triphones, it is called transition matrices tying.

After initialization, several iterations of Baum-Welch reestimation algorithm are used. Models of all triphones, found in the training data, will be the result. These models are not applicable for recognition yet, because of the lack of robustness (some of the triphones could occur in the training data only once). So the next necessary step is state-tying, where all similar states from different models are tied up. With state tying, more data is available for training of each state and hence model robustness is increased. For example, triphones *a-b+s* and *a-b+l* have very similar parameters of their first states, because both are describing the *a-b* transition. So these two states can be tied up, next trained as one state and thus be more robust. After training, a set of tied states, a set of triphones and a set of triphone-to-tied states references are obtained. Each triphone model contains three references to three tied states. Every tied

state can be shared by more triphone models. For state-tying possibilities determination, phonetic binary trees are used. They are based on phonetic questions like: „Is the left context of the phoneme a consonant?“ or „Is the right context of the phoneme vowel *a*?“. Answers to these questions are always only yes, or no, so it is always decidable, if two states can be tied up or not. The state-tying algorithm starts with monophone models and tries to divide states using the binary tree questions into triphones. With state tying, thresholds R0 and TB have to be defined. R0 and TB values affect the degree of tying and therefore the result number of tied states. R0 defines the minimal number of frames that every tied state has to have after division. TB is a minimal logarithmic probability increase, arisen from division of one state onto two. Detailed description of state-tying algorithm can be found in [4].

For triphone models initialization, the following one-mixture monophone models were used:

1. Initialization: Data1_AS
   Reestimation: Data1, 12 iterations
2. Initialization: Data1_MS + Data2_MS_Male
   Reestimation: Data1 + Data2_Male, 12 iterations

For state-tying, the following R0 and TB values were used:

1. R0 = 100, TB = 300 - the most often used combination. The R0 value ensures a sufficient robustness of one-mixture three-state models (100 frames for each Gaussian mixture) and the TB value an adequate probability increase.

2. R0 = 50, TB = 100 - small threshold values result in more tied states, than in the first case, but there will be not enough data for robust model training.

3. R0 = 300, TB = 0 – the maximal number of tied states is needed, regardless the probability increase. Models with 300 frames per state can be later turned into more-mixture models with robustness preserved.

The results of the training are shown in Tab. 2.

Nr	Mix	Training	R0	TB	Triph.	States	Log. prob.
9	1	Data1	100	300	3916	2687	-63,595
10	1	Data1	50	100	5324	6468	-62,535
11	3	Data1	300	0	3721	2056	-62,435
12	1	Data1+ Data2_Male	100	300	9664	27474	-64,208

Tab. 2. Triphone models training results.

3.3 Comparing Models

In our research, 8 sets of monophone and 4 sets of triphone models were trained. Now we have to find the best set, which will be used for automatic segmentation of our data. Average logarithmic probability per frame was the only model quality criterion so far. In this chapter, we will show its reliability.

For our tests, the manually segmented part of Data1 was used. With all models, the automatic segmentation has been done and shifts between manually and automatically segmented boundaries have been measured. In Fig. 3-6, there are histograms that show the frequency of boundary shifts of different lengths. On the x-axis, boundary shifts with 10 ms steps are presented with following rules: All borders with an accuracy error between -5 and 5 ms are included in 0 ms value. All borders with an error between +5 ms and +15 ms are included in +10 ms value. All borders with an error between -5 ms and -15 ms are included in -10 ms value and so on. The y-axis represents the number of incorrect borders to all borders ratio for each 10 ms shift.

From these histograms it is obvious, that the most accurate segmentation was reached with the model set number 3 (32-mixture monophones). Overall 37% of phoneme borders has been shifted less than ±5 ms and 72% of them has been placed into ±15 ms interval. In comparison with common 64-mixture models for continuous speech recognition (Fig. 3), there is more than 10% difference in ±5 ms interval. Other 32-mixture monophone models had very similar results (Fig. 4), models 8 were the worst. In Fig. 5, there are 8-mixture monophone models results. It was proved, that models initialized with automatically segmented data (models2) are much worse, than models initialized with manually segmented data (models1).

Triphone models (Fig. 6), although proposed to be better than monophones, were worse in result. One possible reason could be insufficiency of training data. There were no significant differences between triphone models in their results.

Logarithmic probability has appeared to be a treacherous criterion of model quality. For both 8-mixture and 32-mixture variants, logarithmic probability was higher for models initialized with automatically segmented data. In our practical tests, models initialized with manually segmented data were much better.

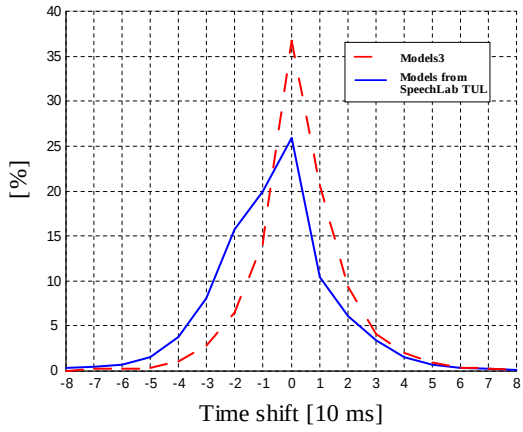

Fig. 3. Best models for automatic segmentation compared with common models for continuous speech recognition.

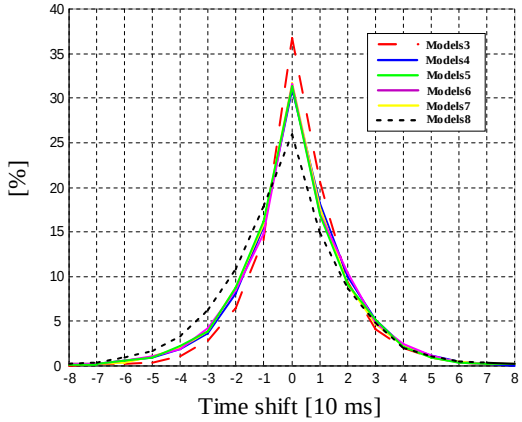

Fig. 4. 32-mixture monophone models.

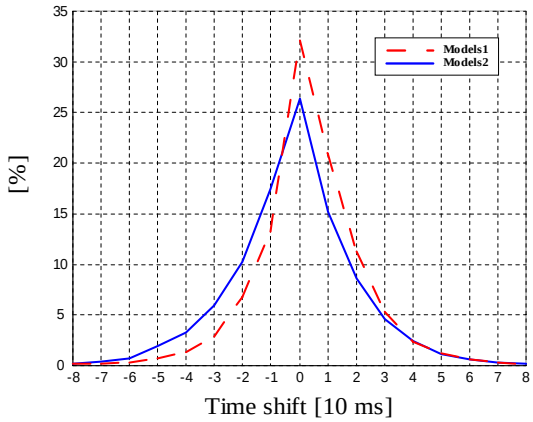

Fig. 5. 8-mixture monophone models.

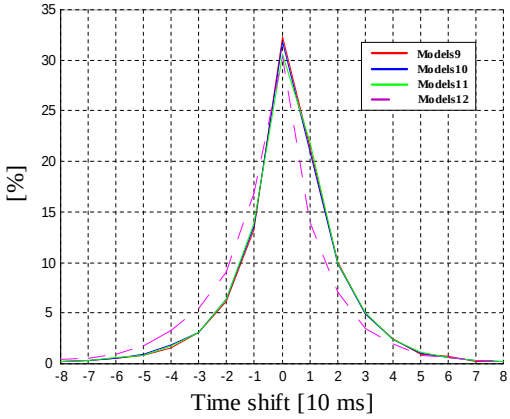

Fig. 6. Triphone models.

4. Conclusion

In this paper we focused on large speech database automatic phoneme segmentation. HMM training process has been discussed, with emphasis on signal framing and speech corpus quality. To prove our statements, several HMM variants has been trained and segmentation tests has been done. Reliability of logarithmic probability as a model quality indicator has been disproved. From our work, the following conclusions have been done:

- In speech framing, the frame rate should be less than 5 ms for accurate segmentation.
- It is necessary to have enough training data (100 frames per mixture) to keep sufficient model robustness.
- Triphone models compared with monophones are harder to train, more computer time is needed and worse results are given.
- For the initialization phase, a small part of manually segmented data is better to use, than all the data automatically (inaccurately) segmented.
- Logarithmic probability is not a reliable model quality indicator.

This method has been used for unit database creation in real triphone-based TTS system [1] with a very satisfactory result.

Acknowledgements

This work has been partly supported by the Grant Agency of the Czech Republic (grant no. 102/05/0278) and by the Grant Agency of the Czech Academy of Sciences (grant no. 1QS108040569).

References

[1] KROUL, M. *Triphone-based speech synthesis*. Diploma thesis, Technical University of Liberec, 2006. (in Czech)

[2] NOUZA, J., MYSLIVEC, M. Methods and application of phonetic label alignment in speech processing tasks. *Radioengineering*, 2000, vol. 9, no. 4, p. 1-7.

[3] HORÁK, P. Automatic speech segmentation based on alignment with a text-to-speech system. In *Improvements in Speech Synthesis.* Ed. Keller, E.; Bailly, G.; Monaghan, A.; Terken, J.; Huckvale, M. Chichester, J. Wiley, 2002, p. 328-338.

[4] MATOUŠEK, J. *Text-to-Speech Synthesis Using Statistical Approach for Automatic Unit-Database Creation.* Dissertation thesis, University of West Bohemia, Pilsen, 2000. (in Czech)

[5] HUANG X., ACERO A., HON H. *Spoken Language Processing*. Prentice Hall, 2001, ISBN 0-13-022616-5.

[6] NOUZA, J. Spectral variation functions applied to acoustic-phonetic segmentation of speech signals. In *Speech Processing (Forum Phoneticum, 63)*, pp. 43-58, 1997.

[7] YOUNG, S., KERSHAW, D., ODELL, J., OLLASON, D., VALCHEV, V., WOODLAND, P. *The HTK Book*, version 2.2. Entropic Ltd., 1999.

About Author...

Martin KROUL was born in 1983 in Liberec (Czech Republic) and has been a PhD. student at the Technical University in Liberec since 2006. He is interested in computer speech recognition and synthesis.