Development of a baseline system for phonemes recognition task

Maros Jakubec, Eva Lieskovska, Roman Jarina, Michal Chmulik, Michal Kuba
Department of Multimedia and Information-Communication Technologies, University of Zilina
Univerzitna 8215/1, 010 26 Zilina, Slovak Republic

Abstract. *The phonemes recognition is one of the fundamental problems in automatic speech recognition. Despite the great progress in speech recognition, discrimination of isolated phonemes is still challenging task due to coarticulation, and great variability in speaking style. The aim of this work is to develop a system for classification of isolated English vowels from the TIMIT dataset. In the paper, the following conventional methods are compared: a) k-Nearest Neighbours approach as a simple nonlinear instance-based classifier b) Gaussian Mixture Model, which belongs to the class of probabilistic acoustical modelling techniques. As a front-end, we applied standard mel-frequency cepstral coefficients with their time derivates. Various experimental methods such as trimming of audio data and cross-validation were used to increase recognition precision and reliability of system evaluation. The developed system will be used as a baseline for comparison with other newer state-of-the-art approaches.*

1 Introduction

Despite the significant progress in automatic speech recognition (ASR) in recent decades, the role of phonemes recognition is still a challenging task. Many experiments have been made to improve the performance of phoneme recognition, including the use of better features or multiple features combinations, improved statistical models, é criteria or modelling of pronunciation, noise, language and more [1].

In the paper, we present an ongoing work on development of the system for classification of isolated English vowels from the TIMIT dataset. The developed system will be used as a baseline for comparison with more advanced state-of-the-art approaches. In the paper we discuss system performance using a) *k*-nearest neighbours (*k*-NN) as a simple nonlinear instance-based classifier, and b) probabilistic approach based on Gaussian Mixture Model (GMM). Speech spectrum is represented by conventional mel-frequency cepstral coefficients (MFCC).

1.1 Related works

Sha and Saul [2] introduced a system for phonemes recognition. They trained GMM for multiway classification, using the basic principle of SVM. With MFCCs including their deltas (time derivates) and 16 Gaussian mixtures they achieved 69.9% accuracy. Deng and Yu [3] used the Hidden Trajectory Model on a phone recognition task. Similarly, feature vectors consist of joint static cepstra and their deltas. The resulting accuracy was 75.17%. Hifny and Renals [4] introduced a phonetic recognition system based on TIMIT database where an acoustic modulation is achieved through augmented conditional random fields. They achieved 73.4% accuracy using the core test set and 77% in test which includes the complete test set. A publication from Mohamed et al. [5] reports the use of neural networks for acoustic modelling. The outcome is 79.3% accuracy in the core test.

The above-mentioned works are focused on different type of phone set from the TIMIT database. Several studies regarding the vowels classification have also been made. Weenink [6] proposed vowel classification improvement by including information about the known speaker into the process. The goal was to reduce the variance in vowel space. The 13 monophthong vowels were selected similarly as in [7]. Linear discriminant analysis on bark-scale filter bank energies was used as a classification method. They reported that information about spectral dynamics improved the classification process. Reduction of the between-speaker variance and the within-speaker variance resulted in higher classification accuracy.

An empirical comparison of five classifiers was presented in [8]. SVM, *k*-NN, Naive Bayes, Quadratic Bayes Normal (QDC) and Nearest Mean algorithms were tested for vowel recognition using the TIMIT Corpus. MFCCs were used for signal parameterization. The results of this experiment show that SVM classifier achieved the best performance. The QDC classifier had the lowest accuracy. The error rate of QDC method has decreased about 10% by using the combination of *k*-NN-QDC-NB. Such combination of classifiers can be efficient way to boost the performance of machine learning method.

Amami et al. [9] conducted a study on different SVM kernels for a multi-class vowel recognition from the TIMIT corpus. Investigation of the optimal parameters of the kernel tricks and the regularization parameter was done. Two different features such as MFCC and PLP were also applied. Middle frames of the vowels and Fuzzy c-means clustering (FCM) were evaluated to determine the appropriate front-end analysis. The method based on middle frames outperforms FCM method. Three middle frames turned out to have the best recognition accuracy. Interestingly, the results showed that the recognition accuracy decreased as the number of frames increased Regarding SVM classification, the accuracy of the vowel system and the runtime improves with smaller value of the kernel width and the regularization parameter.

Palaz et al. [10] claim that the ASR system based on a neural network can be modelled by end-to-end training procedure, without the need of separation into feature extraction and classifier parts. In the proposed method, raw speech waveform was used as an input to the CNN-based speech recognition system. According to the results on the TIMIT phonemes and the Aurora2 connected words recognition tasks, the CNN-based end-to-end system yields better performance than a standard spectral feature extraction-based system.

Although it is not always possible to achieve exactly the same comparison of existing systems, Table 1 summarizes

some of the most important systems in the field of TIMIT phonemes recognition over the last twenty years. Subsequently, the presented survey is ranked according to the system accuracy, including the used methods and the sets of features.

Table 1. Comparison of existing works related to phoneme classification

Authors	Proposed Methods	Descriptors	Classes	Accuracy
Biswas, A. et al. [24]	Hidden Markov Model (HMM)	wavelet based features (84 - PCA)	21 phonemes	88.90 %
Karsmakers P. et al. [13]	SVM- RBF Kernel	181 dimensional	39 phonemes	82.90 %
Mohamed et al. [5]	Monophone Deep Belief Networks	MFCC, Δ, ΔΔ, energy (39)	39 phonemes	79.30 %
Siniscalchi et al. [14]	TRAPs, temporal context division + lattice rescoring	MFCC, Δ, ΔΔ, energy (39)	39 phonemes	79.04 %
Hifny & Renals [4]	HMM	13 MFCC, Δ, ΔΔ, (39)	39 phonemes	77.00 %
Deng & Yu [3]	Hidden Trajectory Models	static / delta cepstra	39 phonemes	75.17 %
Sha & Saul [2]	GMMs trained as SVMs	13 MFCC, Δ, ΔΔ, (39)	39 phonemes	69.90 %
Frejd & Ouni [26]	HMM	13 MFCC, Δ, ΔΔ, - PLP (39)	39 phonemes	67.60 %
Dimitri Palaz et al. [12]	- two-layer MLP - HMM decoder	MFCC, Δ, ΔΔ, energy (39)	39 phonemes	66.65 %
Palaz et al. [10]	-Convolutional neural network - HMM decoder	Raw speech	39 phonemes	65.50 %
Weenink [6]	Linear discriminant analysis	54 dimensional	13 vowels	60.30 %
Amami et al. [8]	SVM- RBF Kernel - middle frames selection	MFCC, Δ, ΔΔ, (36)	20 vowels	51.60 %

2 Proposed methods

2.1 Dataset

The TIMIT Acoustic-Phonetic Continuous Speech Corpus (LDC) database [1, 15] was used for classification. The TIMIT speech corpora contains read speech and is primarily designed for studying acoustic-phonetic phenomena and for testing automatic speech recognition systems. 630 people participated in creating of this database, each contributing by reading 10 phonetically rich sentences. The recordings are in the eight main dialects of American English.

Audio files are recorded at 16 000 Hz, 16 bit. Each audio file is accompanied by metadata files containing phonetic and lexical transcriptions.

2.2 Features extraction methods

The extraction of appropriate features is one of the basic task of objects recognition. In the conventional ASR front-end, speech is represented by a sequence of feature vectors retaining particularly useful information from the signal. There are a large number of approaches and features extraction methods in ASR techniques. The features that have been used in our algorithm will be described in the following section.

Mel Frequency Cepstral Coefficients - are the most commonly used acoustic features in ASR. MFCCs are designed to respect non-linear sound perception by human ear [16].

In our system, the MFCCs are computed as follows (Fig. 1): The pre-emphasis is applied to the speech signal in order to emphasize its high-frequency components. The next step is to divide the signal into 16 ms long frames with an overlap of 1/2 of the frame length. The given frame length was selected based on previous studies on isolated phonemes recognition [8, 11, 24]. The number of signal samples (256) is chosen as power 2 due to the use of FFT. A Hamming window is applied to frames to maintain the continuity of the first and last points in the frames. The signal is converted to the frequency domain by using the FFT algorithm. The magnitude frequency response is then calculated. The spectrum values are multiplied by a series of 20 triangular bandpass filters, summed for individual filters and then logarithmized.

The triangular filter bank has a linear frequency distribution in the Mel frequency range:

$$mel(f) = 1125 * \ln\left(1 + \frac{f}{700}\right) \quad (1)$$

where f [Hz] is the frequency in the linear scale and mel (f) [mel] corresponds to the frequency in the mel scale.

The last step is to calculate the coefficients using the discrete cosine transformation DCT.

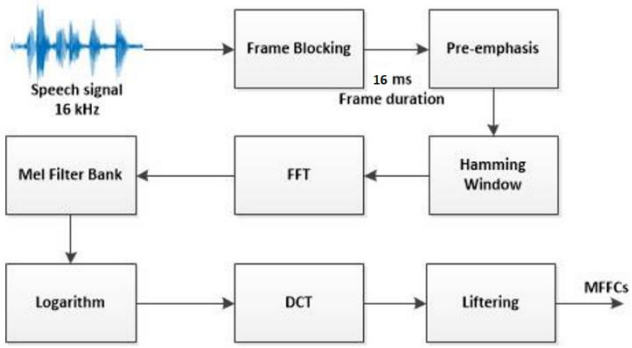

Fig. 1. Block diagram of the MFCC computation

An important parameter is also the energy of the frame. Log energy is usually added as the 13th feature to MFCC. *The short-term energy* function is defined by:

$$E = \sum_{k=-\infty}^{\infty} [\, s(k)\, w(n-k)]^2 \qquad (2)$$

where *s(k)* is signal sample in time *k* and *w(n)* is the corresponding window type. It is then possible to obtain an average energy value for each frame. The disadvantage of this characteristic is the high sensitivity to rapid changes in the signal level. Values of this characteristic can be also used to separate silence segments from speech segments.

Static features, which are obtained using the procedure above, do not capture inter-frame changes along time index. Therefore, dynamic (or delta) features are commonly appended to the feature vectors. Usually delta features are the estimates of the time derivatives of static features and are a computed as follows [17]:

$$\Delta_k[i] = \frac{\sum_{m=1}^{M} m(c_k[i+m] - c_k[i-m])}{\sum_{m=1}^{M} m^2} \qquad (3)$$

where $\Delta_k[i]$ is the delta coefficient, from frame *i*, c_k is the static coefficient and a typical value for *M* is 1.

In the developed system, total features consist of 39 elements per frame:

- 12 MFCC,
- 12 delta (ΔMFCC),
- 12 delta-delta (ΔΔMFCC),
- 3 log energy.

2.3 Classification

The classification process can be divided into a learning and testing phase. Thus, data set needs to be divided into two subsets. Because of 10-fold cross-validation evaluation process (2.4), we selected the same number of vowels from each class.

Once the data were split, models of selected vowels were trained and tested according to the chosen method. The general classification scheme can be seen in Fig. 2.

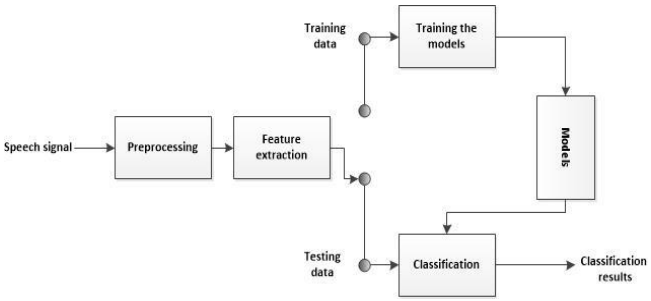

Fig. 2. Block diagram for classification scheme

There are several methods suitable for phoneme classification task addressed in this work. The following well-established classifiers, namely Gaussian mixture model (GMM), Gaussian mixture model-Universal background model (GMM-UBM) and a *k*-nearest neighbours (*k*-NN), were chosen for the baseline system development due to their easy implementation and good classification properties. We recall a description of these methods in the following section.

The Gaussian Mixture Model works on the principle of probabilistic modelling of audio features in the feature space. GMM is defined as the probability density function formed by a linear superposition of *K* Gaussian components [18][19] as follows:

$$p(x) = \sum_{k=1}^{K} \pi_k N\,(\boldsymbol{x}|\boldsymbol{\mu}_k, \sum_k) \qquad (4)$$

where, the probability density function of the multivariate Gaussian distribution for *n*-dimensional vector $\boldsymbol{x}$ is given by:

$$N(\boldsymbol{x}|\mu, \sum) = \frac{1}{(2\pi)^{\frac{n}{2}} |\sum|^{1/2}} \exp\,(-\frac{1}{2}(\boldsymbol{x}-\mu)^T \sum^{-1} (\boldsymbol{x}-\mu)) \qquad (5)$$

with mean vector $\boldsymbol{\mu} \in \mathbf{R}^n$ and covariance matrix $\sum \in \mathbf{R}^{n \times n}$. π_k are mixing coefficients, which must satisfy the following conditions

$$0 \le \pi_k \le 1 \;\text{ and }\; \sum_{k=1}^{K} \pi_k = 1 \qquad (6)$$

The classification function for the proposed GMM classifier has the following form:

$$f(x) = arg \max_{C}\; ({}^{C}p(\boldsymbol{x})) \qquad (7)$$

where ${}^{C}p(\boldsymbol{x})$ is GMM of the class C.
Thus, we are looking for the maximal probability over all *C* classes.

The training algorithm, which returns a set of parameters $\Theta = \{\boldsymbol{\mu}, \Sigma \text{ and } \boldsymbol{\pi}\}$ for each class, is based on the Maximum Likelihood (ML) criterion. Given the model $p\,(\boldsymbol{x}, \Theta)$ with the unknow parameters, the aim is to derive its parameters based the training data – set of the feature vectors $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$. The ML method uses Fisher likelihood function, which is defined as:

$$F(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N | \boldsymbol{\theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n | \boldsymbol{\theta}) \qquad (8)$$

The maximum of this function with respect to unknown parameters Θ can be formalized as follows:

$$\hat{\theta} = \arg\max_{\theta} \sum_{n=1}^{N} \log p\,(\boldsymbol{x}_n | \boldsymbol{\theta}) \qquad (9)$$

The maximization defined by (9) is a complicated task that does not have an explicit solution. The *expectation-maximization* (EM) algorithm [18] is used for finding maximum likelihood solutions.

Training the GMM statistical model for each single vowel is challenging for both computing power and memory. Fitting the model also suffers from lack of a sufficient amount of training data. It is therefore advisable to train a universal generic model (so called *Universal Background Model* UBM), which represents the possible distribution of the features for a wide group of sounds, and then derive from

it the class-specific model for an individual vowel. The Maximum likelihood estimation (ML) of the model parameters is used for UBM training [20].

The Maximum a posteriori probability (MAP) estimate is used for UBM adaptation to the vowel model (i.e. class-specific GMM). In the presented experiments, only vectors of mean values of UBM were adjusted to obtain individual models.

Given a sequence of features vectors $\boldsymbol{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_N\}$ from one class of vowels, the score is expressed by (10), where $\boldsymbol{\theta}_v$ and $\boldsymbol{\theta}_{UBM}$ denote the actual vowel model and universal model respectively. According to (10), the greater the probability $p(\boldsymbol{o}_n|\boldsymbol{\theta}_v)$ against background model for as many feature vectors as possible, the more will be supported the hypothesis that the recognized audio sample belongs to the given vowel class.

$$score = \frac{1}{N}\sum_{n=1}^{N}\log\frac{p(\boldsymbol{o}_n \mid \boldsymbol{\theta}_s)}{p(\boldsymbol{o}_n \mid \boldsymbol{\theta}_{UBM})} \tag{10}$$

The k-Nearest Neighbours (*k*-NN) is a simple nonlinear instance-based classification method and is one of the most popular classical approaches of cluster analysis. It classifies an unknown sample based on the known classification of its neighbours [21][22].

The model itself is essentially made up of a training set, and the learning process consists in storing of patterns from all training samples in one model. Given an unknown sample, the distances between the unknown sample and all the samples in the training set can be computed. Input attributes must be numeric so that their distance can be calculated for each of the two patterns. Samples from the training set have n number attributes, and each one sample represents a point in the N-dimensional space. If a classifier wants to determine the target attribute of an unknown sample, it searches in the k sample space of the training set for those that are closest to that unknown sample. Training set can be defined as:

$$\{\boldsymbol{x}_i, C_i\}_{i=1,\ldots,K},\ C_i \in \{1,2,\ldots,L\} \tag{11}$$

where $\boldsymbol{x}_i$ is a sample with its corresponding label C and K is the size of the whole training set, L is a number of classes (i.e. number of vowels). Given unknow sample x, we are looking for sample x_k according to following formula:

$$\|\boldsymbol{x}_k - \boldsymbol{x}\| = min\|\boldsymbol{x}_i - \boldsymbol{x}\|_{i=1,\ldots,K} \tag{12}$$

Subsequently, the sample $\boldsymbol{x}$ is placed to the same class that $\boldsymbol{x}_k$ belongs to.

In the proposed system we used the Euclidean distance, which is the most commonly used metric for distance determination, as well as the city-block, Chebyshev and cosine distance metrics. They are defined as follows:

$$d_{Euclidean}(\boldsymbol{x},\boldsymbol{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{13}$$

$$d_{city-block}(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{n}|x_i - y_i| \tag{14}$$

$$d_{Chebyshev}(\boldsymbol{x},\boldsymbol{y}) = \max_i(|x_i - y_i|) \tag{15}$$

$$d_{cosine}(\boldsymbol{x},\boldsymbol{y}) = \frac{\sum_{i=1}^{n}x_i y_i}{\sqrt{\sum x_i^2}\sqrt{\sum y_i^2}} \tag{16}$$

From the above-mentioned facts it's obvious that two important factors play a role in the successful classification:

- the choice of distance function
- the choice of the value for the parameter k (i.e. number of neighbours)

It is advised to choose an odd number for k to avoid the scenario when two classes labels achieve the same score. Some issues need to be considered during the selection of k value. Classes with a great number of samples can overwhelm small ones and the results will be biased, so it is not recommended to set large k value. The advantage of using many samples in the training set is not exploited if k is too small [21].

The disadvantage of this classifier is the calculation of all distances for each classification, which can considerably slow down the process and it can be computationally expensive if the training set or the number of unknown samples is large.

2.4 *k*-fold cross-validation

If there is not a sufficient number of observations, an appropriate approach to determine the optimal solution for training/testing is the so-called cross-validation technique. [23].

The data set is divided into k parts, with one part always being used for testing, and the remaining k-1 parts being used for training. The process is repeated so that each part is used for testing just once (Fig. 3). The advantage of validation is a relatively accurate estimate of the classification success. The disadvantage of validation is that it requires more computer memory and consumes more time because a lot of calculations are needed at every step.

Fig. 3. *k*-fold cross-validation

3 Experimental setup and results

The evaluation of the proposed GMM, GMM-UBM and *k*-NN methods was performed. All the tests were evaluated on isolated vowels extracted from the TIMIT data set. Two

sets of vowels were created. The first set consists of the 5 classes *aa*, *eh*, *iy*, *ow*, *uh*. This subset correlates with the common vowels of the most European languages (e.g. 'a','e','i','o','u' in Slovak) [25]. The second set consists of 18 American English vowels (see Table 4 for a list). The set of the 5 classes was used in the first and second experiments. Finally, performance of developed system was evaluated on the second set of the 18 classes.

Proposed algorithms were implemented in *MATLAB 2018b* with support of the *Voicebox* [27] and *Netlab* [28] toolboxes.

Classifier training and testing was performed by 10-fold cross-validation. Data was initially randomly divided into 10 equally large subsets. Each of them contained approximately the same number of vowels represented by the feature vectors. Nine of them were used to train the model and the rest one to test it. This was repeated 10 times, so that all 10 subsets were tested. All data were parameterized by 39 MFCCs (incl. deltas and delta-deltas) per 16 ms frame with 8 ms overlap. The features matrix dimension for each vowel was 10800x39 (frames x features).

The results of the experiments with 5 vowels classification using *k*-NN and simple trained GMM are shown in Tables 2 and 3 respectively. There are shown the results achieved for various *k*-NN setup (type of metric and number of neighbours) and GMMs (number of gaussians and covariance matrix types) settings. An effort has been made to achieve a better classification accuracy by editing the data. Therefore, the entire database was mixed so that the speech dialects are evenly distributed between the training and the test part. Another data modification was vowel trimming by omitting the first and last frames for each vowel recording. So that silent parts as well as parts affected by coarticulation or unprecise vowel border detection were not taken into account. In addition, the middle frames are known to contain the most important information about the vowel. Such modified data are referred as D_2, D_1 indicates original data.

Table 2. The overall system accuracy for 5 vowels recognition, using *k*-NN classifier, and 2 data manipulation techniques: whole vowels (D1), trimmed vowels (D2)

Metric	k=3		k=5		k=7	
	D_1	D_2	D_1	D_2	D_1	D_2
Chebyshev	73.54	89.48	74.53	86.61	74.81	84.37
Cosine	74.56	91.24	75.62	88.57	75.94	86.23
Euclidean	75.83	92.13	77.33	90.25	77.87	88.67
Cityblock	75.64	95.08	79.47	92.96	79.80	91.19

Table 3. The overall system accuracy for 5 vowels recognition, using GMM classifier, and 2 data manipulation techniques: whole vowels (D1), trimmed vowels (D2)

Covartype	n=16		n=32		n=64	
	D_1	D_2	D_1	D_2	D_1	D_2
ppca	80.42	81.64	79.28	80.37	78.21	80.86
diag	82.53	84.73	83.47	85.93	84.85	85.84
full	86.53	87.45	83.80	91.10	82.33	86.25

Significant improvement can be seen for both methods of classification if only stationary middle part of the vowels is analysed (D2). At *k*-NN method, a success rate of 95.08% with k = 3 neighbours and cityblock metric, was achieved. GMM achieved the best success rate of 91.1% at n = 32 gaussians and full covariance matrix. The comparison of the best results for 5 vowels achieved by the above-mentioned methods is shown in Fig. 4.

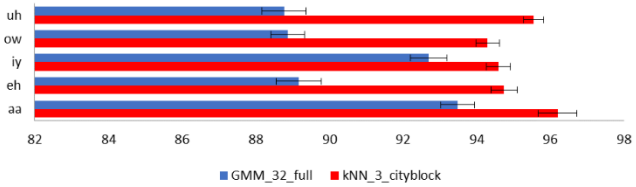

Fig. 4. The comparison of classification of 5 selected vowels

In the last experiment, testing was performed on a larger set of classes - 18 vowels of American English were selected. Data needed for UBM training were selected from other recordings available in the database. A total of 4600 recordings from 510 speakers in a total length of approximately 3 hours and 54 minutes were used to train the UBM model. The front-end with data manipulation is the same as in experiments with the recognition of 5 vowels (referred as D2 in the text above). The experiments with GMM-UBM training/classification approach is also added. Fig. 5 shows the best results achieved Interestingly, the *k*-NN algorithm outperformed both GMM and GMM-UBM approaches. It achieved 84.2% vowel recognition accuracy, at setting k = 5 neighbours and cityblock metric. The second most successful system was GMM-UBM, which achieved success rate of 78.1% at n = 256 gaussians and full covariance matrix. The worst performance had the GMM classifier, probably due to insufficient amount of training data It achieved a system success rate of 75.5% at n = 16 gaussians and full covariance matrix.

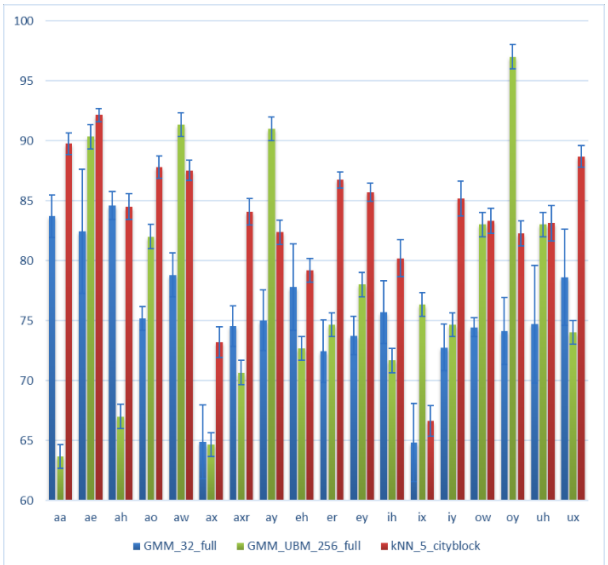

Fig. 5. The comparison of classification of 18 vowels

Table 4 shows the classification of the individual vowels for the best *k*-NN model settings in form of confusion matrix. The data in table indicates the performance of the algorithm as well as the false recognized vowels. This is the best way to see how the system works when recognizing individual vowels. The diagonal shows the correctly classified vowels. The lines specify incorrectly identified vowels. The final success rate in percentage is also stated.

	aa	ae	ah	ao	aw	ax	axr	ay	eh	er	ey	ih	ix	iy	ow	oy	uh	ux
aa	270	5	12	12	8	7	3	6	6	5	0	0	0	0	3	6	0	0
ae	6	279	6	6	4	8	3	8	9	0	4	3	6	0	3	5	0	0
ah	5	0	255	5	5	18	3	6	6	5	3	3	5	0	9	6	3	0
ao	3	0	3	265	4	9	0	2	3	0	0	0	0	0	6	9	3	0
aw	4	0	3	3	265	6	0	3	4	0	0	0	0	0	4	3	0	0
ax	3	0	6	3	3	224	3	2	5	3	3	3	9	3	9	6	9	3
axr	3	0	0	0	0	3	255	2	3	18	3	3	7	3	0	0	3	2
ay	4	4	3	0	3	3	3	251	2	0	3	2	3	4	0	3	0	0
eh	2	6	3	0	5	3	0	3	240	3	6	6	7	3	3	3	3	0
er	0	0	0	0	0	0	19	0	0	261	0	3	3	0	0	0	3	0
ey	0	3	3	0	0	0	0	6	5	0	259	9	12	9	0	3	0	3
ih	0	0	0	0	0	0	0	3	5	0	3	244	15	9	3	3	6	4
ix	0	0	0	0	0	3	3	3	3	0	2	6	203	6	3	3	3	6
iy	0	3	0	0	0	1	2	2	3	2	12	13	15	257	0	0	4	13
ow	0	0	3	3	0	3	0	1	3	0	0	0	0	0	251	0	3	0
oy	0	0	3	3	3	3	0	2	3	0	0	0	0	0	3	250	3	0
uh	0	0	0	0	0	6	3	0	0	0	0	2	3	0	3	0	252	2
ux	0	0	0	0	0	3	3	0	0	3	2	3	12	6	0	0	5	267
%	89,80	92,71	84,49	88,33	88,33	74,19	84,49	83,62	79,81	86,73	86,39	81,19	67,61	85,68	83,62	82,27	84,12	89,80

Table 4. Confusion matrix of phoneme recognition for the best *k*-NN model

The total number of correctly classified vowels was 4548 out of 5400 and the success rate of 84.2% was achieved. As seen from Fig. 5 and Table 4, in the case of *k*-NN, the vowels: *aa*, *ae*, *ao*, *aw*, and *ux* were recognized best, while for the vowels *ax*, *eh*, and *ix*, a considerable number of samples were misclassified. Note that using GMM-UBM classifier, largest recognition errors occurred in other group of vowels (see Fig. 5). The largest difference in recognition rate between *k*-NN and GMM-UBM is in the case of the vowels *aa*, *ux*, *ix*. From Fig 5, also disbalance between simple GMM and GMM-UBM can be seen (theoretically, GMM-UBM should outperform GMM in all cases). Probably, further optimization of GMM-UBM is required.

Phoneme recognition task on the TIMIT database consists of several years of intensive research. There exists a number of systems and their classification success has naturally improved over time. Results presented in this paper are comparable to the existing research reported in the literature (see section 1.1). However, it is not possible to compare these works directly with our system because of different parameters and experimental settings that have been used.

4 Conclusion

This work deals with the design of a system for recognition of isolated vowels extracted from the TIMIT dataset and subsequent optimization of the training algorithm. Three different approaches for phoneme classification were *k*-NN, GMM, and GMM-UBM. The *k*-NN method achieved the best results with overall accuracy of 95.08% for 5 vowels and 84.2% for 18 vowels recognition. GMM-UBM gave comparable results for 18 vowels recognition but classification error was distributed differently among vowel classes than in the case of *k*-NN. This recognition disbalance issue between *k*-NN and GMM approaches needs further investigation.

Acknowledgment

This publication is the result of the project implementation: Centre of excellence for systems and services of intelligent transport II, ITMS 26220120050 supported by the Research & Development Operational Programme funded by the ERDF.

References

[1] C. Lopes, F. Perdigao, Phone recognition on the TIMIT database. Speech Technologies, IntechOpen 2011, pp. 285-302.

[2] F. Sha, L. K. Saul, Large margin Gaussia nmixture modelling for phonetic classification and recognition. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, 2006 (ICASSP), France, May 2006.

[3] L. Deng, D. Yu, Use of differential cepstra as acoustic features in hidden trajectory modelling for phonetic recognition. Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2007.

[4] Y. Hifny, S. Renals, Speech recognition using augmented conditional random fields. IEEE Transactions on Audio, Speech & Language Processing, vol. 17, no. 2, 2009, pp. 354–365, ISSN 1558-7916. 2009.

[5] A. Mohamed, G. Dahl, G. Hinton, Acoustic Modeling using Deep Belief Networks", IEEE Transactions on Audio, Speech, and Language Processing1558-7916, 2011.

[6] D. Weenink, Vowels normalizations with the TIMIT acoustic phonetic speech corpus. Institute of Phonetic Sciences, University of Amsterdam, Proceedings 24, 117–123, 2001.

[7] H.M. Meng, V.W. Zue, “Signal representation comparison for phonetic classification”, in IEEE Proc. ICASSP, Toronto, 285–288, 1991.
[8] R. Amami, D.B. Ayed, N. Ellouze, An Empirical Comparison of SVM and Some Supervised Learning Algorithms for Vowel recognition. In: International Journal of Intelligent Information Processing, 2012.
[9] R. Amami, D.B. Ayed, N. Ellouze. Practical selection of svm supervised parameters with different feature representations for vowel recognition. Int J Digit Content Technol Appl, 7/2013, pp. 418-424.
[10] D. Palaz, M. Magimai.-Doss, R. Collobert, Analysis of CNN-based Speech Recognition System using Raw Speech as Input. In Proceedings of the 16th Annual Conference of International Speech Communication Association (Interspeech), Dresden, Germany, 6–10 Sept. 2015; pp. 11–15.
[11] O. Farooq and S. Datta, Phoneme recognition using wavelet based features, Information Sciences 150, 2003, pp. 5-15.
[12] D. Palaz, R. Collobert, M. Magimai.-Doss, End-to-end Phoneme Sequence Recognition using Convolutional Neural Networks. Idiap, Dec. 2013
[13] P. Karsmakers, K. Pelckmans, J. Suykens, H. Van Hamme, Fixed size kernel logistic regression for phone classification. Proceedings of Interspeech 2007, 1990-9772 Belgium, 2007.
[14] S.M. Siniscalchi, P. Schwarz, C.H. Lee, High-accuracy phone recognition by combining high-performance lattice generation and knowledge based rescoring. Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, 2007.
[15] S. J. Garofolo, L.F. Lamel, W.M. Fisher, J.G. Fiscus, D.S. Pallett, N.L. Dahlgren, V. Zue, TIMIT Acoustic-Phonetic Continuous Speech Corpus. Linguistic Data Consortium, Philadelphia, 1993.
[16] R. Jang, Audio Signal Processing and Recognition: 12-2 MFCC (2005), (available at: http://mirlab.org/jang/books/audiosignalprocessing/speechFeatureMfcc.asp?title=12-2%20MFCC).
[17] S. Young, et al., “The HTK Book (for HTK Version 3.4),” Cambridge University Engineering Department, 2006.
[18] Chuong B. Do. “The Multivariate Gaussian Distribution.” Stanford, CA, USA, 2008.
[19] Ch. M. Bishop, Pattern Recognition and Machine Learning. Springer, 2006.
[20] A. R. Avilla, S. P. Milton, F. J. Fraga, D. D. O’Shaughnessy, T. H. Falk, Improving the Performance of Far-Field Speaker Verification Using Multi-Condition Training: The Case of GMM-UBM and i-vector Systems. In: Proceedings of the Fifteenth Annual Conference of the International Speech Communication Association. Singapore, 2014
[21] A. Mucherino, P.J. Papajorgji, P.M. Pardalos, Data mining in agriculture. Springer Dordrecht Heidelberg London New York, ISBN 978-0-387-88614-5 pp. 83-8, 2009.
[22] P. Cunningham, S.J. Delany, *k-Nearest neighbour classifiers*. Technical Report UCD-CSI-2007-4, Dublin: Artificial Intelligence Group, 2007.
[23] Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation, Journal of Machine Learning Research, 5:1089–1105, 2004.
[24] A. Biswas, P.K. Sahu, A. Bhowmick, M. Chandra, Feature extraction technique using ERB like wavelet sub-band periodic and aperiodic decomposition for TIMIT phoneme recognition. International Journal of Speech Technology, Volume 17, Issue 4, pp 389–399, December 2014.
[25] P. Grzybek and M. Rusko, Letter, Grapheme and (Allo-)Phone Frequencies: The Case of Slovak, Glottotheory, vol. 2, No. 1, 2009, pp 30–48.
[26] I. Ben Fredj and K. Ouni, Optimization of Features Parameters for HMM Phoneme Recognition of TIMIT Corpus, International Journal of Advanced Research in Electrical, Vol. 4, Issue 8, Aug. 2015.
[27] M. Brookes, VOICEBOX: A speech processing toolbox for MATLAB (available at http://www. ee. ic. ac. uk/... hp/staff/dmb/voicebox/voicebox. html).
[23]I. Nabney, Netlab: Pattern analysis toolbox (available at https://www.mathworks.com/matlabcentral/fileexchange/2654-netlab).