Süleyman Demirel Üniversitesi
Fen Bilimleri Enstitüsü Dergisi
Cilt 22, Özel Sayı, 319-329, 2018

Süleyman Demirel University
Journal of Natural and Applied Sciences
Volume 22, Special Issue, 319-329, 2018

DOI: 10.19113/sdufbed.12798

Turkish Speech Recognition Based On Deep Neural Networks

Ussen Abre KIMANUKA*[1], Osman BUYUK[1]

[1]Kocaeli University, Faculty of Sciences, Department of Electronic and Communications Engineering,41380, Kocaeli

(Alınış / Received: 28.12.2017, Kabul / Accepted: 17.07.2018, Online Yayınlanma / Published Online: 05.09.2018)

Keywords
Turkish speech recognition,
Deep neural network,
Gaussian mixture model,
Hidden Markov model,
GMM-HMM,
DNN-HMM

Abstract: In this paper we develop a Turkish speech recognition (SR) system using deep neural networks and compare it with the previous state-of-the-art traditional Gaussian mixture model-hidden Markov model (GMM-HMM) method using the same Turkish speech dataset and the same large vocabulary Turkish corpus. Nowadays most SR systems deployed worldwide and particularly in Turkey use Hidden Markov Models to deal with the speech temporal variations. Gaussian mixture models are used to estimate the amount at which each state of each HMM fits a short frame of coefficients which is the representation of an acoustic input. A deep neural network consisting of feed-forward neural network is another way to estimate the fit; this neural network takes as input several frames of coefficients and gives as output posterior probabilities over HMM states. It has been shown that the use of deep neural networks can outperform the traditional GMM-HMM in other languages such as English and German. The fact that Turkish language is an agglutinative language and the lack of a huge amount of speech data complicate the design of a performant SR system. By making use of deep neural networks we will obviously improve the performance but still we will not achieve better result than English language due to the difference in the availability of speech data. We present various architectural and training techniques for the Turkish DNN-based models. The models are tested using a Turkish database collected from mobile devices. In the experiments, we observe that the Turkish DNN-HMM system have decreased the word error rate approximately 2.5% when compared to the GMM-HMM traditional system.

Derin Sinir Ağlarına Dayalı Türkçe Ses Tanıma

Anahtar Kelimeler
Türkçe ses tanıma,
Derin sinir ağı,
Gauss karışımı model,
Saklı Markov modeli,
GKM-SMM,
DSA-SMM

Özet: Bu çalışmada derin sinir ağları (DSA) kullanılarak Türkçe konuşma tanıma sistemi geliştirilmiş ve bu sistemle elde edilen sonuçlar geleneksel Gauss karışım model-saklı Markov modeli (GKM-SMM) yöntemi ile aynı ses ve geniş dağarcıklı metin veri tabanı kullanılarak karşılaştırılmıştır. Günümüzde dünyanın birçok bölgesinde ve özellikle Türkiye'de kullanılan konuşma tanıma sistemlerinde konuşmadaki zamansal değişimleri modellemek için saklı Markov modelleri tercih edilmektedir. Akustik verinin gösterimi olan öznitelik katsayılarına her bir SMM durumunun hizalama olasılıklarını tahmin etmek için Gauss karışım modeli kullanılmaktadır. Hizalama olasılıklarını tahmin etmek için kullanılabilecek bir diğer yöntem de ileri-beslemeli derin sinir ağlarıdır; bu DSA birkaç pencereden oluşan öznitelik katsayılarını girdi olarak alıp, HMM durum sonsal olasılıklarını çıktı olarak vermektedir. Özellikle İngilizce, Almanca gibi dillerde DSA'ların geleneksel GKM-SMM yöntemine göre daha başarılı sonuçlar verdiği gösterilmiştir. Türkçe'nin eklemeli bir dil olması ve Türkçe için hazırlanmış geniş ses veri tabanlarının bulunmaması DSA ile yüksek performanslı konuşma tanıma sistemlerinin gerçekleştirilmesini güçleştirmektedir. Bu çalışmada, İngilizce gibi geniş veri tabanlarının bulunduğu dillerdeki kadar olmasa da Türkçe bir ses tanıma sisteminin performansının DSA ile arttırılabileceği gösterilmiştir. Çalışmamızda, Türkçe için farklı DSA mimarileri ve eğitim yöntemleri ile sonuçlar sunulmuştur. Testler mobil akıllı telefonlardan alınmış kayıtlardan oluşan Türkçe bir veri tabanında gerçekleştirilmiştir. Deneylerde, önerilen DNN-HMM sisteminin kelime hata oranını geleneksel GMM-HMM yöntemine göre yaklaşık %2.5 oranında azalttığı gözlenmiştir.

* *Corresponding author: ukimanuka@gmail.com*

1. Introduction

Comparing the performance of speech recognition products commercially deployed in the world and particularly in Turkey with human level performance, we realize that these automatic speech recognition systems (ASR) are still far behind the human level perspective [1, 2].

The improvement of discriminative training in speech recognition (see in [3]; such as: maximum mutual information (MMI) estimation [4], boosted MMI [9], minimum classification error (MCE) training [5, 6] and minimum phone error (MPE) training [7, 8] provided better result for traditional GMM-HMM systems. However despite these improvements, the accuracy of ASR was still far behind the human level performance. This implied more research were required.

The first use of neural networks in speech application in 1990s never showed outstanding performance compared to the traditional GMM-HMM technology [10]. They were some key problems experienced in neural network such as the vanishing gradient, lack of big training data, lack of big computation power and weak temporal correlation structure in the neural predictive models [11, 12]. In 2010s, the benefit of developing deep models was started with the idea of resolving these limitations. Among these resolutions (1) the addition of recurrence to the DNN resulting in the dynamics of speech being approximately captured by the standard HMM; (2) reversing the direction of information flow from top-down used for generative modeling to bottom-up used in DNN; (3), development of efficient initialization techniques based on deep belief network (DBN) or pre-training [13] since training of neural network possessing many hidden layers was found to be difficult. The adoption of these three resolutions changed the traditional neural network into the DNN-based ASR framework.

In the past eight years, DNN have shown beyond doubt to be performant in many domain of research such as image processing, speech recognition, language modeling, parsing, information retrieval, speech synthesis, speech translation, cars automation, gaming, etc. When deep learning is applied on these domains, the results obtained bypass the state-of-the-art results. The reason for this high performance of DNN is their capability to find and learn very compound structure in large amount of data. These deep neural nets have been used in speech recognition to develop acoustic models derived from this deep learning technology. In [14] architectures of a DNN-HMM hybrid are proposed for speech recognition on broadcast news dataset and the performances obtained were very satisfactory.

DNN techniques have been mainly implemented for popular languages such as: English, Chinese, and Spanish. Speech recognition (SR) research for the Turkish language is mostly concentrated on language modeling since the agglutinative nature of Turkish results in high out of vocabulary (OOV) rate in large vocabulary tasks and this degrades the recognition performance. In [15, 16]), sub-word units such as syllables or stem-endings are proposed to alleviate the OOV problem. In [17], vowel harmony rule in Turkish is incorporated to SR. In [18], long short-term memory (LSTM) and compositional LSTM recurrent neural networks are used for Turkish broadcast news transcription task. In [19], a speaker independent and phone based continuous time digit sequence recognition system was designed. The system used the traditional HMM for acoustic modeling. HMM was trained using 50 speakers and showed approximately 74% recognition rate. In [20], an isolated syllable based SR system was designed using neural networks. . The vocabulary of the designed system is limited to 50 Turkish words. In [21], word and syllable-based Turkish SR systems are developed using dynamic time warping (DTW), multilayer perceptron (MLP), support vector machine (SVM) and HMM methods. All the developed systems are middle-sized and user-dependent.From these researches we have realized that the use of deep neural networks for acoustic modeling has not been fully exploited for Turkish large vocabulary continuous speech recognition (LVCSR), yet. In this paper, we develop a Turkish LVCSR system using deep neural networks and compare it with the previous state-of-the-art traditional Gaussian mixture model-hidden Markov model (GMM-HMM) method. For this purpose, we constructed a Turkish speech database which consists of recordings from mobile smart phones since there is no publicly available database for the task. We chose the mobile SR domain since it is an emerging application field [22]. In the future, we plan to distribute the database for academic research purpose.

In this paper, we propose a hybrid setup of deep neural network (DNN) and Hidden Markov model (HMM) on a subset of Turkish speech dataset [22]. The neural network is trained using 6.1hours of Turkish speech obtained by mobile-recordings of 549 Turkish native speakers. We will denote our system trained with deep neural network with the abbreviation DNN-HMM and the traditional Gaussian Mixture model will be abbreviated as GMM-HMM. In this paper, we illustrate the procedures and the steps involved to develop DNN-HMM and analyze how different choices in the design can exert influence on the recognition power, and we clearly show that DNN-HMM can be more profitable than the traditionally-trained GMM-HMM baseline on our mobile recorded dataset [22].

Section II of this paper presents an overview of automatic speech recognition. Section III introduces deep neural networks. In section IV, the developed system and the experimental evaluation are clearly explained. Then in section V we give a conclusion and future work.

2. Overview of Automatic Speech Recognition

Speech recognition usually deals with the difficulty of determining what the message in a portion of audio signal. It is also known as the operation of extracting linguistic information from speech signals. A simple automatic speech recognition is made up of three main components [23]: (1) *an acoustic mode*l (which gives the properties of the sounds of the language). (2) *a phonetic lexicon* (this gives the words accompanied with their pronunciations). (3) *a language model* (which has some knowledge in the word sequences that can be uttered). Obtaining these three components requires statistical operations to be applied on a very large amount of audio data and text corpora.

In ASR we make use of an acoustic model, language model and lexicon to obtain the best sequence of words $\widehat{W}$ given a recorded speech X. Mathematically, automatic speech recognition can be represented by the following expression:

$$\widehat{W} = argmax_w \, P(X|W)P(W) \qquad (1)$$

Where $\widehat{W}$ is the greatest word sequence that makes as large or great as possible the above likelihood;

$P(X|W)$, represent the *acoustic probability* (in simple terms it is the likelihood of an audio signal X given the word sequence W);

$P(X|W)$, is calculated by means of acoustic model;

$P(W)$, represent the language probability (it the *a priori* likelihood of word string, calculated by means of the language model).

2.1. Acoustic modeling

The statistical representations of different sounds are computed by acoustic modeling. Acoustic modeling is usually done with the help of Hidden Markov Model (HMM). A statistical model within which the modeled structure is presumed to be a Markov procedure with not observed or hidden states is known as an HMM [24]. Speech is usually modelled using Hidden Markov Models since it has the capability to handle temporal evolutions in data and it has been the core framework for speech modelling since their first application to speech recognition.

HMM is basically a stochastic finite state machine containing N number of states, and having mainly three constituents: $\{A, B, \pi\}$. Training of HMM is done by calculating the parameter set A, B, π. Where A is the transition probability matrix; B is emission probability vector (with $b_j(x)$ the emitted likelihood of observation x found in state j); π is prior probability vector (π_i the prior likelihood of state i).

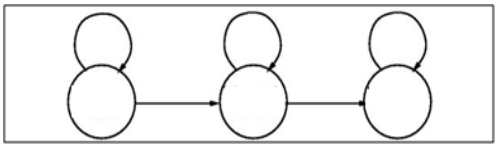

Figure 1. Shows an HMM composed of 3 states, it topology is directed from left-to-right and possess self-loops; a typical example of the HMM used in speech application

HMM is basically a stochastic finite state machine containing N number of states, and having mainly three constituents: $\{A, B, \pi\}$. Training of HMM is done by calculating the parameter set A, B, π. Where A is the transition probability matrix; B is emission probability vector (with $b_j(x)$ the emitted likelihood of observation x found in state j); π is prior probability vector (π_i the prior likelihood of state i).

Before 2012, the parameter B was estimated by means of Gaussian mixture distribution to obtain $\{\mu, \Sigma, C\}$ which represent respectively the means, the covariance and weights of the mixtures. This emission probability is mathematically represented by the following function:

$$b_j(x) = \sum_{m=1}^{M} c_{jm} \, N(x; \, \mu_{jm}, \Sigma_{jm}) \qquad (2)$$

This parameter is estimated recursively by the Baum-Welch Algorithm.

The use of HMM in speech modeling is because of its potential to make provision for the dynamic aspect of speech, in simple terms whether a person speak slowly or quickly, this model will still be able to recognize the speech.

In automatic speech recognition, the modelling of sounds in a language (referred to as a phone) is accomplished by means of a three-state HMM (see Figure 1). The role of these three states is to catch the beginning, the center and the end sections of a speech (phone). For context-independent phone models, the best capturing of co-articulation is performed by tri-phone models.

Detailed explanation on HMM can be found here [24].

2.2. Language modeling

Language models give a large catalogue of words and their respective likelihood of occurrence in a sentence (in simple terms it aims at computing the probability of sequence of words). The popular

method for modeling a language is the model known as statistical *n-gram* (in which *n-gram* model provides a likelihood of a given word w_i in relation to the $n-1$ antecedent words). The language model is represented mathematically as follow:

$$P(w_i|w_1,\dots,w_{i-1}) = P(w_i|w_{i-(n-1)},w_{i-(n-2)}\ \dots,w_{i-1}) \quad (3)$$

Language modeling requires a very large text corpus in order to estimate these probabilities. The commonly used language modeling toolkits are SRILM [25] and MITLM [26].

Language Modeling toolkit can be defined as a collection of tools developed for the appropriate calculation of statistical n-gram language models with iterative parameter estimation (see in [25, 26] for more information).

2.3. Lexicon

A lexicon is a set of files that enable you to identify the pronunciations for words. It can also be defined as a dictionary of pronunciations. These words and their respective pronunciations give a connection between language models and sub-word HMMs. The pronunciations found in a lexicon are created based on phones.

2.4. Search or decoding

Search or decoding consists of an algorithm that determines the word sequence $\widehat{W}$. Since the search space is too large, in order to find good solution recognition is performed in two distinct steps:

(1) The first stage aims at removing words possessing a low probability and then constructs a lattice utilizing beam search. This lattice is made by hypotheses of best word. The hypotheses are made up of words, time boundaries of the words, acoustic probabilities and language model probabilities.

(2) This stage involves browsing the lattice by means of some information to produce the greatest hypothesis.

The metric used to measure the achievement of automatic speech recognition is known as WER (word error rate). It usually make a comparison between the reference and the hypothesis obtained by the search. It can be mathematically represented by the following expression:

$$WER = \frac{S+D+I}{N} \quad (4)$$

Where S represents the number of substituted words, D representing the number of deleted words, I the number of inserted words and N the number of words found in the reference.

3. Deep Neural Networks

In the field of speech recognition, deep neural networks are considered as a hot topic. It was introduced in 2012 and many companies around the world (such as Google, Microsoft, etc.) are starting to use this technology in their production systems.

Before the introduction of deep neural network, Gaussian mixture models were used for building a map between sub-phonetic states and audio input features. These GMM systems had the disadvantage of producing a speech recognition system containing a large number of sub-components, linguistic assumptions, and approximately extremely huge amount of code [27]. Then, in the year 2012 DNN were introduced in order to play the role that was previously performed by the Gaussian Mixture models. In present time, DNN are popular in many type of applications because of their capability to perform a superior level of supposition on very large amount of data by means of a deep graph containing linear and non-linear transformations (see Figure 2). The deployment of deep neural network technology was possible because of the advancement in hardware; specifically their training has been possible with the help of GPU (Graphical Processing Units).

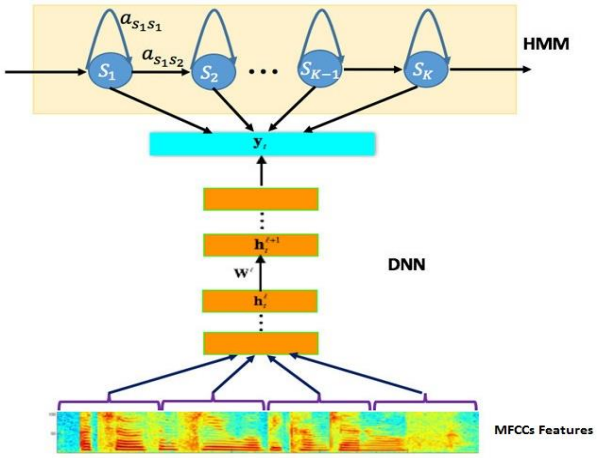

Figure 2. Schematic diagram of a DNN-HMM model (figure extracted from [28]). The sequential property of the speech is modelled by the HMM and DNN models the scaled observation likelihood of all tied tri-phone (Senones)

Deep neural networks consist of interconnected neurons. These neurons are structured into layers. These layers are subdivided into 3 layers: the input layer (consisting of data features), the hidden layer (consisting of more than 1 hidden unit), and lastly the output layer (which handle the classification task by giving out the probabilities of labels). The DNN have more than one hidden layer that is the reason why it is mentioned as *deep*. Depth property of a neural network is very important since many researches

have shown that the use of many hidden layers result in an improved WER and simple training procedures. So (according to [29]) discussion on how to obtain good performance for deep networks are investigated by many researchers, but many have agreed that the deeper networks are better compared to the classical neural networks. DNN example is given in Figure 3 as follow:

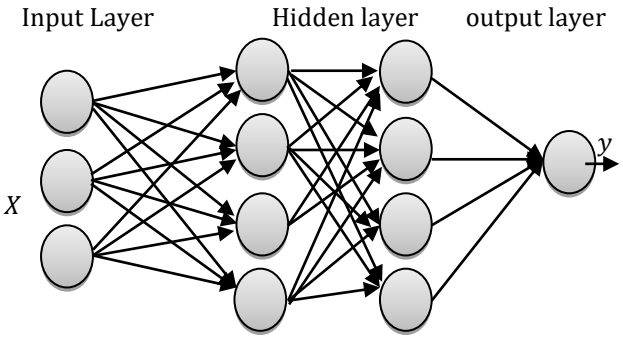

Figure 3. Example of a DNN having three neurons in it input layers and two Hidden layers of 4 four neurons each and one output neuron in the output layer. This is known as deep because there are more than one hidden layer

The activation of a neuron in a DNN can be mathematically represented by the following expression:

$$y = f(\sum_{i=1}^{n} w_i x_i + b) \qquad (5)$$

There are three parameters that can define a DNN: (1) the interconnections found between the different layers; (2) the process of updating the weights w_i known as training process; (3) the activation function f (which help in the conversion of the weighted input of the neuron to its output activation).

The commonly used activation function is the non-linear activation functions because they can outperform the linear counterpart which can only separate linear classes. The most used non-linear functions are: Logistic sigmoid function, Rectified Linear Units, maxout, and hyperbolic tangent function (tanh). There are two types of generalized maxout activation function: the P-norm and the soft max generalized maxout function. In this paper the hyperbolic tangent function (in Figure 4) and the generalized maxout P-norm function are commonly used in the experimentation.

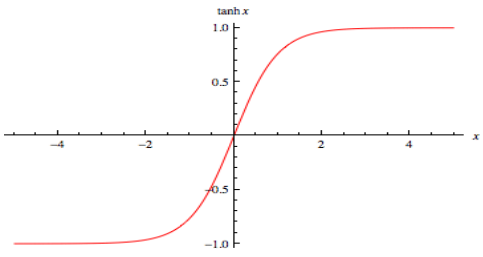

Figure 4. Hyperbolic tangent activation function used in DNN [14]

3.1. Training deep neural networks

The purpose of training a DNN is to decrease the error between the calculated outputs and the desired values. This technique aims at estimating the weights w_i of all the neurons in the network. Here training is made by utilizing a classical learning algorithm known as gradient descent. At every epoch the following operation is performed:

$$\Delta w = -\eta\left(\frac{dE}{dw}\right) \qquad (6)$$

Where E is the cost function: the difference found between the desired output and the calculated output. η Is the learning rate (this gradually decreases with training).

The above mentioned expression in (6) is computed to calibrate the weights w so that the error found between the desired output and the computed output is reduced at a small value [30]. This gradient is computed on a large training dataset. See in [31] for detailed procedures involved in the process of DNN training.

3.2. Types of DNN

In terms of their architecture, there are four types of DNN:

(1) *MLP (Multilayer Perceptron)*: in which the neuron found in each layer is joined to all the other neurons found in the preceding or antecedent layer. It also known as Feedforward neural network.
(2) *RNN (Recurrent Neural Network)*: this network is able to compute it output at time t based on the information computed at a previous time (*t-1*).
(3) *LSTM (Long Short-Term Memory)*: this is a particular RNN which tries to improve the gradient vanishing and memory problem which were affecting the RNN. It was developed by Sepp Hochreiter and Jürgen Schmidhuber [32].
(4) *CNN (Convolutional Neural Network)*: this fall in particular feedforward neural network that are very efficient for speech application.

It should be important to mention that all these types of neural networks can be used to train an acoustic model in speech recognition. But researches have proven RNN and LSTM to possess advantage over the other types. RNN and LSTM have got the capability to account for temporal evolution of the input features.

3.3. DNN parameters

One of the difficulties faced in DNN is the selection of choice of parameters. There are many parameters involved when tuning the neural network training, the most important parameters are: *the amount of*

hidden layers, *the hidden layer dimension*, *selection of the non-linear functions*, and *choosing a learning rate adjustment function*.

These parameters are chosen based on the problem involved, the size of the dataset and the scarcity of data. There are many criterion on selecting the parameters of a DNN, in our case we used informal trials to determine the parameters (clearly explained in section 4.5).

3.4. Acoustic modeling using DNN

Acoustic modeling using DNN is quite different from the GMM acoustic modeling. In the traditional GMM-HMM training, the observation probability $b_j(x)$ of each HMM state was calculated using Gaussian mixtures but in the case of DNN-based acoustic model this observation probability of each HMM phone state is calculated using DNN given the audio signal [33]. In DNN-based acoustic model we are taking audio attributes and allocate a class to those attributes (i.e. a phoneme label). This simply means our acoustic model will possess input consisting of the dimensions of our acoustic characteristic (i.e. 39 input nodes for 39 Mel-frequency cepstral coefficients), and output nodes consisting of senone labels (i.e. 900 output nodes for 900 context dependent triphones (decision tree leaves)).

Technically, when training a GMM-HMM system, monophone model are trained from utterance-level transcriptions. But in the case of DNN-HMM, training is not done from utterance-level transcriptions, it starts from phoneme-to-audio alignments level which are usually produced by the traditional GMM-HMM system. This insinuates that in order to train a DNN you will have to obtain the phoneme-to-audio alignments produced by the traditional GMM-HMM system. So the acoustic modeling of DNN depends on the attributes used to train a GMM-HMM as well as the decision tree produced by the GMM-HMM.

The acoustic feature frames are inserted into the first layer (input layer), the job of the neural network will be assigning a phoneme label to each acoustic frame, and due to the fact that we possess the phoneme label from GMM-HMM alignments of every frame, we just make a comparison between the neural network estimated phoneme and the real phoneme (GMM-HMM). The above process is summarized in Table 1. By means of a loss function and backpropagation, we repeat over all the frames used for training example to make adjustment in the weights and the biases of the neural network [34].

Table 1. DNN Input and output features description

Input features (x)	MFCCs
Output (y)	Phoneme label to a frame(senone labels)
Desired output (d)	Phoneme label from our GMM-HMM alignments
Training	Difference between desired output and the output computed by the neural net. Then update the weights and the biases

3.5. Language modeling using DNN

Some recent research in [35] have shown that classical N-gram used for language modeling has a weakness in generalization (this insinuates that when a string of words has not been seen at training process, the N-gram model will produce a poor estimated probability). So to solve the problem, neural networks have been used to account for the temporal structure of language. The commonly used neural networks for Language modeling are LSTM and RNN. In this paper we are not training our language model using DNN. This will be addressed in future work.

4. Experiments

This section explains in details the development of our Turkish speech recognition system using deep neural network. Our system is built using a speech recognition toolkit known as *Kaldi toolkit* [36] .This toolkit is available for free under the Apache License.

4.1. Dataset/Corpus

In this paper, we perform experiments on 6.1 hours of Turkish recorded speech, these speeches were recorded using telephones (smartphones) for Turkish speech recognition purpose (database explained in [22] and summarized in Table 2). This speech database contains 549 speakers including males and females. The audio was recorded from mobile platform using pulse code modulation with mono channel and sampled at 16 kHz, 16 bit. For language modelling (LM) a collection of 396k Turkish sentences obtained from the internet were used to build our unigram and bi-gram language model [22].

Table 2. Information related to the Turkish Dataset [22]

	Duration in Hours	***TotalNumber of utterances***
Training Set	4.8 hours	4025
Test set	1.3 hours	1157

4.2. Parametrization

We are sampling our audio signal at 16 KHz. For speech signal analysis we use 25ms frames having a switch of 10ms. 13 Mel-frequency cepstral

coefficients are computed for each respective frame completely by the 13 delta and 13 delta-delta coefficients resulting to an observation vector of 39-dimension.

4.3. Acoustic models

As mentioned early in this paper, the simple comparison between GMM-HMM and DNN-HMM acoustic modeling, reside in the computation of $b_j(x)$ (emission probability). We trained two systems baseline system which uses Gaussian mixtures GMM-HMM and a DNN-HMM system. While training the acoustic model for GMM and DNN the language modeling and the lexicon remain unchanged. The traditional GMM acoustic modeling used 11k Gaussians. And the DNN is a MLP with 3 or 5 hidden layers of 375 or 750 hidden layer dimension (neurons per layer). The selection of hidden layer size and dimension was done by informal trials explained in section 4.5.

4.4. Lexicon and language models

- ***Lexicon***

Turkish language have a Latin alphabet which is a little bit similar to the English alphabet, but the only difference is the addition of some non-Latin vowels such as Ü, Ö, I and non-Latin consonants such Ç, Ş, Ğ. Adding these special letter into our Kaldi toolkit [36] is impossible, so the commonly used techniques for Turkish lexicon Building is to replace the non-Latin vowels and consonants by a capitalization system that allow distinction between the Latin and the non-Latin letter. First of all at the utterance level transcription the non-Latin vowels and consonants were replace by a capitalization format (see in Table 3) and the this was continued up to lexicon and dictionary level (see in Table 4).

Table 3. Sample of utterance level transcription for Turkish language

Original sentence in Turkish	***Utterance level transcription in Kaldi***
Sağlığımız için bakliyat fazla tüketmeliyiz	saGlIGImIz iCin bakliyat fazla tUketmeliyiz
Kızlar küçükken kendilerine papatyadan taçlar yaparlar	kIzlar kUCUkken kendilerine papatyadan taClar yaparlar
Güllerin içerisinden en gösterişlisi kırmızı olanlarıdır	gUllerin iCerisinden en gOsteriSlisi kIrmIzI olanlarIdIr

Table 3 shows the representation of these non-Latin letters at utterance level transcription. As we can see the letters Ü,Ö,I, Ç,Ş,Ğ are represented respectively by the capital letters U,O,I,C,S,G at utterance level transcription. Kaldi toolkit [32] is able to make a distinction between these capitalized letters to the lower case letters. This capitalization system is commonly used in Turkish utterance transcription.

At lexicon and dictionary level these non-Latin letters are represented differently, to allow differentiation in the pronunciation.

Table 4. Sample of lexicon representation for Turkish language

Original sentence in Turkish	***Utterance level transcription in Kaldi***	***Lexicon (word's pronunciation)***
aç	aC	a C1
ağustos	aGustos	a G1 u s t o s
böyle	bOylece	b O1 y l e c e
broşür	broSUr	b r o S1 U1 r

Table 4 shows the representation of the non-Latin letters at utterance level transcription and at lexicon or dictionary level.

Using this transcription system helped Kaldi toolkit [36] to differentiate the non-Latin alphabet from the Latin alphabet and our lexicon was successfully implemented. Our lexicon contains 3759 Turkish words with their relevant Turkish pronunciations.

- ***Language model***

Our Turkish language model was trained from a very large text corpus accumulated from the internet, particularly including Turkish newspaper corpus. The overall size was 1.26 million words. Our n-gram language model is obtained using the Language model toolkit (explained in [22]) on each text corpus. In this article, unigram (1-gram) and bigrams (2-gram) model are used with 1.2 million unigram and 18 million bigrams as well as a lexicon of 3759 words with their relative pronunciations.

4.4. Training and decoding

Kaldi speech recognition toolkit [36] was exploited for building our Turkish speech recognition systems. The other libraries used in deep learning are compared in [37].

Firstly, we trained our GMM-HMM system as follow: the acoustic modeling is created using 13 Mel-frequency cepstral coefficients and Gaussian mixture models on the training data. In addition to that we implement delta-delta coefficients on the MFCCs using linear discriminant analysis (LDA) transformation and also speaker adaptive training (SAT) is performed using maximum likelihood linear regression transform (abbreviated as fMLLR) which is calculated per speaker. The language model was trained on our 1.26 million words Turkish web data using MITLM toolkit [26]. This will result in a traditional GMM-HMM system shown in Table 6.

Secondly, we replace the GMM with deep neural network then the alignments obtained in GMM where

used in the training of our deep neural net as follow: From the MFCCs files inputted into the neural network a Phoneme label to a frame (HMM state) was estimated then stochastic Gradient Descent (SGD) was applied to compare the estimated phoneme label with HMM state obtained from GMM system and then the weights of the neural network where adjusted. To speed up the training of our neural network we computed the gradient on a small portion of training data known as mini-batch then we updated the weights soon after every mini-batch. The reason why we used Gradient Descent algorithm in the training process of our experiment is because it is the cheapest and simplest algorithm available, apart from gradient descent there are other expensive algorithms such as Levenberg-Marquart, Newton's method or Quasi-Newton method, etc. In Table 5, different types of neural networks algorithms are compared. Our network used 5 hidden layers containing each 375 neurons. In our experiment we used either a hyperbolic tangent activation function or a P-norm generalized maxout function to simplify the training process. Since we trained our DNN system using CPU processor we used a mini-batch of 128, so that the obtained result do not differ with the result obtained using a GPU processor. The difference between these two processors resides in the speed of training. By using the CPU, training as well as decoding takes a long time compared to when GPU is used.

Table 5. Comparison of neural network algorithms [38]

Neural network algorithm	Details
Fruit fly optimization algorithm(FOA)	This is a swarm intelligence optimization algorithm which does not utilize the gradient of the optimized problem. It does not assure that an optimal solution is ever found.
Differential Evolution algorithm (DE)	Also used to solve numerical optimization problems. DE do not assure an optimal solution is ever found.
Gradient Descent(GD)	Is the simplest and cheapest training algorithm (used here in our experiment)
Newton method	This algorithm makes use of the Hessian matrix which makes it computationally expensive.
Quasi-Newton	It is the intermediate between gradient descent and Newton's method which is also expensive than Gradient descent.
Levenberg-Marquardt	Works with the gradient vector and Jacobian matrix and requires a lot of memory

4.5. Results

The results of our Turkish recognition are expressed in terms of a metric know as word error rate for different experiments given in Table 6, 7 and 8.

Here two systems have been compared. The two different systems use similar lexicon and language models but their acoustic models are different. In all the shows, the DNN-HMM performs better than the GMM-based system. There is an absolute WER difference of 2.5%. This clearly proof that acoustic modeling based on DNN for Turkish language achieves better performance because of their better classification and their better ability in generalization. Additional reasons for the better performance of DNN-HMM are: (1) GMM requires de-correlation processing for input characteristics, but DNN is capable of using various forms of input characteristics; (2) GMM can only use single-frame speech as inputs, but DNN is capable of capturing valid context information by means of splicing adjoining frames.

Table 6. GMM-HMM comparison with DNN-HMM using the same language model and lexicon

Turkish ASR system	*WER (%)*
GMM-HMM (using fMLLR features)	17.40
GMM-HMM(using MMI discriminative training)	16.77
GMM-HMM (using fMMI discriminative training)	16.32
DNN-HMM(using TANH)	14.72
DNN-HMM(using P-NORM)	14.65

We also compared the performance of our Turkish DNN while the total model size and the numbers of the hidden layers are varied. According to the experiment a DNN with unique hidden layer perform very poorly in contrast with DNNs having 2, 3 or 5 hidden layers. But increasing the hidden layer to a value of 7 did not increase the performance. So there is no benefit from a 7 hidden layer model when compared with a 2, 3 or a 5 hidden layer model. It was observed that the depth of a deep neural net has a very big effect on the performance than total DNN size. That is the reason why it is advised to pick a suitable *amount of hidden layers* than it is to pick a suitable *total model size* (By Model size we mean the number of neurons per hidden unit). Also when a DNN is tremendously deep it becomes more laborious to diminish it training objective function.

Theoretically deeper DNNs should have the capability to model very complex functions than simple neural network, but practically it was observed that the depth can behave like a regulator because of the complexities in optimization for extremely deep networks.

Apart from the number of hidden layers and the hidden layer size, we have another important parameter: the learning rate. Our DNN-HMM used an initial learning rate and a final learning rate of 0.02 and 0.004 respectively. These two values were selected based on [39]. It is generally advised that

the final learning rate should be one fifth or one tenth of the initial learning rate. Also the size of the database matters a lot, for a large database of more than 3.1 hours such as ours, making the learning rate higher will result in a longer training. For database having hundreds of hours, the learning rate can be ten times smaller than the default value (0.04). However for our database of 6.1 hours selecting 0.02 perfectly satisfied our training.

Another parameter for our DNN-HMM is the minibatch size. According to [39] it should be chosen a power of two such as 128, 256 or 512. Having a large minibatch size is considered to be helpful since it assumes the interaction with optimizations utilized in matrix multiplication code, especially when GPU is used. In case of CPU usage, a large minibatch size can lead to instability. Because of these reasons, minibatch size is set to 128 for multi-threaded CPU based training, it is set to 512 for GPU based training. Since our DNN-HMM used a multi-threaded CPU, minibatch size of 128 was selected.

Max-change parameter should be chosen with respect to minibatch size. This implies that as the minibatch increases the max-change also increases. In our experiments, a small max-change was used.

According to [39], one rule should be followed to select the number of epochs: perform the training for a sample number of epochs (num-epochs) which reduces the learning rate from initial to final learning rate and then keep the final learning rate at a fixed rate for extra number of epochs (num-epochs-extra). Therefore, in total num-epochs + num-epochs-extra epochs are performed. For a large database it is required to use small epochs (15+5). Since our database was small, we trained for more epochs (20+5).

Table 7. Comparison of Turkish DNNs using different frameworks such as amount of hidden layers, amount of neurons in each layers, here we are using hyperbolic tangent activation function

Turkish ASR system(6.1h)	*Number of Hidden layer*	*Hidden layer dimension*	*WER (%)*
DNN-HMM	2	750	15.82
	2	375	14.90
	3	750	15.82
	3	375	14.73
	5	375	14.72
	7	375	14.81

Table 8. Difference between HYPERBOLIC TANGENT and P-NORM nonlinearity for Kaldi DNN

Turkish ASR system(6.1h)	*Hidden layer*	*WER (%)*
DNN-HMM using TANH	5	14.72
DNN-HMM using P NORM	2	14.65
	5	14.69

According to our experiments, using the p-norm non-linearity outperformed the hyperbolic tangent activation function in Kaldi speech recognition system (see in Table 8). The difference shown in Table 6 and 8 can be clearly illustrated by the graph in Figure 5:

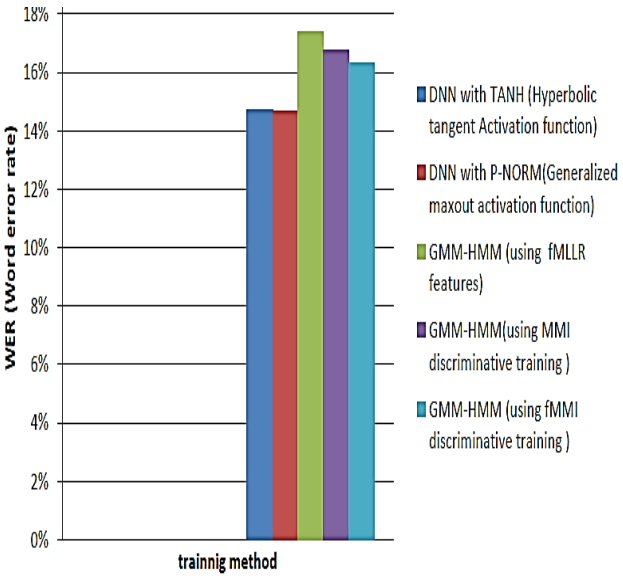

Figure 5. Performance comparison of GMM-HMM and DNN-HMM

To avoid overfitting in our DNN-HMM, cross-validation is used. This technic separate the data in many independent part and leave apart one of these part for testing while the remaining part are used for training. Hence every time we change the test data, we have to retrain the network otherwise the network will be aware of the test data.

Additional testing of our DNN-HMM system was performed on real world speech recognition applications. We tested our system on mobile application such as short message and general text dictation application as in [22], the DNN-HMM provided recognition improvement of 3.81% for short message application meanwhile for the general text dictation an improvement of 4.74% was observed. The testing results are illustrated in Table 9 and Table 10.

Table 9. Short message application results for GMM-HMM and DNN-HMM

Turkish ASR system	*WER (%)*
GMM-HMM	35.07
DNN-HMM	31.26

Table 10. General text dictation application results for GMM-HMM and DNN-HMM

Turkish ASR system	*WER (%)*
GMM-HMM	13.21%
DNN-HMM	8.47%

5. Discussion and Conclusion

We have described deep neural network model for Turkish language that gives better results than the traditional GMM-HMM and also gives results that can compete with the discriminatively trained GMM-HMM on a mobile recorded Turkish dataset. Even though the results provides improvement; it is relevant to indicate that the DNN-HMM training is quite costly in terms of hardware since it requires a GPU processor or many parallelized computer.

One of the drawback of the proposed method is the relatively long training and decoding time, due to parallelized CPU usage in the experiment setup. This can be overcome by using recent GPU processor hence minimizing the training and decoding time.

We observed that an increase in model size and depth can improve WER performance, but there is a certain limit. We also realized that the total network size, not depth can be considered as a critical factor. Possessing more than one hidden layer is of course important but the differences created by DNNs with huge amount of hidden units were seen to be slightly small in contrast to all calculated standard of measurement.

The lack of a very large training corpus should also be mentioned because having large corpus may help in the achievement of a clear gain.

According to [34], when using DNN-HMM the overall error rate reduction in English language vary from 8-20% depending on the amount of data. Since in English language there are thousands of hours of data the performance is way far better. In our experiment the error rate is reduce to a value of 2.5% for only 6.1 hours of Turkish database, this proof clearly that DNN-HMM techniques are able to reduce the error rate compared to GMM-HMM for Turkish language.

We have strong believe that this work on Turkish DNN will open door for more research on Turkish acoustic and language model because many problems remain to be solved. Here are some: First, the designing of algorithms for speaker and environment adaptation in DNN-HMM; second, designing a language model using deep neural network; Third, creation of a very large Turkish corpus (Kaldi recipe), which will help the researchers to perform their experiments. All these will be discussed in our future experimentation.

References

[1] Baker, J.M., Glass, J., Khudanpur, S., Lee, C.H., Morgan, N., O'Shaugnessy, D. 2009. Research developments and directions in speech recognition and understanding, part 1. IEEE Signal Processing Magazine, vol. 26, no. 3, 75–80.

[2] Baker, J.M., Glass, J., Khudanpur, S., Lee, C.H., Morgan, N., O'Shaugnessy, D. 2009. Research developments and directions in speech recognition and understanding, part 2. IEEE Signal Processing Magazine, vol. 26, no. 4, 78–85.

[3] He, X., Deng, L., Chou., W. 2008. Discriminative learning in sequential pattern recognition. IEEE Signal Processing Magazine, vol. 25, no.5, 14– 36.

[4] Valtchev, V., Young, S. J., Kapadia, S. 1993. MMI training for continuous phoneme recognition on the TIMIT database. In Proc. ICASSP, vol.2, 491–494.

[5] Juang, B. H., Hou, W., Lee, C.H. 1997. Minimum classification error rate methods for speech recognition. IEEE Transactions on Speech and Audio Processing, vol. 5, no. 3, 257–265.

[6] McDermott, E., Nakamura, A., Hazen, T.J. 2007. Discriminative training for large vocabulary speech recognition using minimum classification error. IEEE Transactions on Speech and Audio Processing, vol. 15, no. 1, 203–223.

[7] Povey, D., Woodland, P. 2002. Minimum phone error and i-smoothing for improved discriminative training. In Proc. ICASSP, vol. 1, 105–108.

[8] Povey, D. 2003. Discriminative training for large vocabulary speech recognition Ph.D. dissertation, Cambridge University Engineering Dept, 13-21.

[9] Povey, D., Kanesvsky, D., Kingsbury, B., Ramabhadran, B., Saon, G., Visweswariah, K. 2008. Boosted MMI for model and feature space discriminative training. In Proc. ICASSP, 4057–4060.

[10] Deng, L. 2016. Deep learning: from speech recognition to language and multimodal processing. APSIPA, vol. 5, 2.

[11] Bengio, Y., Simard, P., Frasconi, P. 1994. Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw., vol.5, 157–166.

[12] Deng, L., Hassanein, K., Elmasry, M. 1994. Analysis of correlation structure for a neural predictive model with application to speech Recognition. Neural Networks, vol. 7, no. 2, 331-339.

[13] Hinton, G.E., Osindero, S., Teh, Y.W. 2006. A fast learning algorithm for deep belief nets. Neural Computation, vol. 18, 1527–1554.

[14] Mella, O., Fohr, D., Illina, I. 2017. New paradigm in Speech Recognition : Deep Neural Networks. IEEE International Conference on Information Systems and Economic Intelligence, 1-8.

[15] Arisoy, E. 2004. Turkish Dictation System for Radiology and Broadcast News Applications. Msc. Thesis, Bogazici University, 1-5.

[16] Buyuk, O. 2005. Sub-word Language Modelling for Turkish Speech Recognition. Msc. Thesis, Sabanci University, 16-23.

[17] Erdogan, H., Buyuk, O., Oflazer, K. 2005. Incorporating language constraints in sub-word based speech recognition. Automatic Speech Recognition and Understanding, IEEE Workshop on, 98-103.

[18] Arisoy, E., Saraclar, M. 2016. Compositional Neural Network Language Models for Agglutinative Languages. INTERSPEECH, 3494-3498.

[19] Tunca, A. 2010. Digit Sequence Recognition Using Hidden Markov Models and Continuous Speech Recognition Technique. MSc. Thesis, Eskişehir Osmangazi University,20-25.

[20] Urgun, K. 2012. An isolated word syllable based speech recognition system using ANN. MSc. Thesis, Atilim University,9-10.

[21] Özlem, Y. 2016. A comparison of word and syllable-based speech recognition systems. MSc Thesis, Adnan Menderes University, 4-6.

[22] Buyuk, O. 2016. A new database for Turkish speech recognition on mobile devices and initial speech recognition results using the database. Pamukkale University Journal of Engineering Sciences, 1-5.

[23] Deng, L., Li, X. 2013. Machine Learning Paradigms for Speech Recognition. IEEE Transactions on Audio, Speech and Language Processing, vol. 21, n. 5, 1060-1089.

[24] Rabiner, L. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, Proceedings of the IEEE, vol. 77, no. 2, 257-86.

[25] Stolcke, A. 2002. SRILM - An Extensible Language Modeling Toolkit", in Proc. Intl. Conf. Spoken Language Processing,1-4.

[26] Hsu, B.J., Glass, J. 2008. Iterative Language Model Estimation :Efficient Data Structure & Algorithms. InProc. Interspeech, 1-4.

[27] Renals, S. 2017. Automatic Speech Recognition ASR Lecture 11:lexicon and pronunciations, 1-4.

[28] Dahl, G.E., Yu, D., Deng, L., Acero, A. 2012. Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. IEEE transactıons on audio, speech, and language processing, vol. 20, no. 1, 35-36.

[29] Hinton, G., Deng, L., Dahl, G., Mohamed, A., Jaitly, N., Senoir, A., Vanhoucke, V., Nguyen, P., Sainath, T., Kingsbury, B. 2012. Deep Neural Networks for Acoustic Modeling in Speech Recognition,IEEE Signal Processing Magazine, vol. 82, 1-22.

[30] Ba, J.L., Kingma, D.P. 2009. ADAM: A Method for Stochastic Optimization. ICLR, 2-15.

[31] Hinton, G.E., Nair, V. 2009. 3-d object recognition with deep belief nets. in Advances in Neural Information Processing Systems 22, 1339–1347.

[32] Schmidhuber, J., Hochreiter, S. 1997. Long Short-Term Memory, Neural Computation, vol. 9 no. 8, 1735–1780.

[33] Josh Meyer, F. 2016. http://jrmeyer.github.io/ (visited on : 22/11/2017).

[34] Qi, P., Maas, A.L., Xie, Z., Hannun, A.Y., Lengerich, C.T., Jurafsky, D., Ng, A.Y. 2015. Building DNN Acoustic Models for Large Vocabulary Speech Recognition. Computer Speech and Language 41, 195–213.

[35] Chelba, C., Norouzi, M., Bengio, S. 2017. N-gram language modeling using recurrent neural network estimation. Google Tech Report , 1-4.

[36] Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., Vesely, K. 2011. The Kaldi Speech Recognition Toolkit. IEEE Automatic Speech Recognition and Understanding Workshop(ASRU) in Hawaii, US, 1-4.

[37] İnik, Ö., Ülker, E. 2017. Data Sets and Software Libraries Used for Deep Learning, Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT).1-6.

[38] Cömert, Z., Kocamaz, A.F. 2017. A study of artificial neural network training algorithms for classification of cardiotocography signals. Bitlis Eren University journal of science and technology vol.7 no.2, 93-103.

[39] Povey, D., Zhang, X., Khudanpur, S. 2015. Parallel training of DNNs with natural gradient parameter averaging. ICLR, 1-12.