

Ministry of Education and Science of the Republic of Kazakhstan
Suleyman Demirel University

UDC: 004:811.512.122

on manuscript rights

Kuanyshbay Darkhan Nurgazyuly

**Development of methods, algorithms of machine learning and mobile applications
for Kazakh speech recognition**

Thesis

Presented in Partial Fulfillment for the
Degree of Doctor of Science in Computing Systems and Software
(Degree code: 6D070400)
Department of Computer Sciences
Faculty of Engineering and Natural Sciences

Supervisor: prof. Amirgaliyev Y.N
External supervisor: Head of Department of
National University of Ukraine named after
T.Shevchenko, prof. Krak Y.V.

Kaskelen, 2021

CONTENT

NORMATIVE REFERENCES	5
DEFINITIONS	6
LIST OF ABBREVIATIONS	7
1 INTRODUCTION	8
1.1 Problem statement	8
1.2 Aims and Objectives	8
1.3 Novelty	9
1.4 Thesis Outline	9
2 SPEECH SIGNAL PROCESSING	10
2.1 Speech enhancement	10
2.2 Types of noise	11
2.2.1 Wide band noise	11
2.2.2 Removal of interfering speeches	11
2.2.3 Periodic noises	12
2.3 Speech enhancement techniques	12
2.3.1 Single channel enhancement	12
2.3.2 Spectral subtraction technique	12
2.3.3 Spectral subtraction with over subtraction model	13
2.3.4 Non-linear spectral subtraction	13
2.4 Multi-channel enhancement methods	14
2.4.1 Adaptive noise cancellation (ANC)	14
2.4.2 Beamforming	14
2.4.3 Wiener filtering	14
2.4.4 Kalman filtering	15
2.4.5 Linear predictive coding	16
2.4.6 DFT based method	16
2.4.7 Signal subspace method	16
3 ANALYSIS OF EXISTING ASR SYSTEMS	17
3.1 Hidden Markov model based architecture	17
3.2 Feature extraction	18
3.2.1 Linear predictive coding (LPC)	19
3.2.2 Mel-frequency ceptrum coefficients (MFCC)	19
3.3 Acoustic modeling using HMM	21
3.4 Acoustic modeling using ANN	23
3.5 Acoustic modeling using DBNN	25
3.6 Language models in ASR	29
3.6.1 N-gram language model	29
3.6.2 Basic neural network based LM	31
3.6.3 Feed forward neural network language model	31

3.6.4 Recurrent neural network language model (RNNLM)	33
3.6.5 Long Short term memory RNN Language model	34
4 END-TO-END BASED ASR SYSTEMS	36
4.1 CTC-based model	37
4.1.1 The core idea behind CTC	38
4.1.2 Path probability estimation	38
4.1.3 Path aggregation	39
4.1.4 Related works on CTC	41
4.1.5 Training a large data	42
4.2 RNN-transducer based end-to-end model	43
4.2.1 Main ideas behind RNN-transducer	43
4.2.2 RNN-transducer related works	46
4.3 Attention based model	47
4.3.1 Redundancy of the information and delay	48
4.3.2 The structure of a network	48
4.3.3 Related works on Attention based model	49
4.3.4 The problem of continuity	49
4.3.5 Monotonic problem	50
4.3.6 Key information extraction problem	51
4.3.7 Delay	52
4.3.8 Decoder	54
4.4 Summary	55
5 SPEECH CORPORA IN KAZAKH LANGUAGE	57
5.1 Introduction	57
5.2 Related work	58
5.3 Methodology	60
5.3.1 Regulations	60
5.3.2 Architecture	61
5.3.3 Technology	61
5.4 Speech synthesis of Kazakh language application	62
5.4.1 Speech synthesis	63
5.4.1.1 Linguistics	63
5.4.1.2 Prosodic	63
5.4.1.3 Phonetics	63
5.4.1.4 Acoustics	64
5.4.2 Implementations	64
5.4.2.1 Initialization of main parameters	64
5.4.2.2 Text processing	65
5.4.2.3 Sound	65
5.4.3 Neural network training	67
5.4.4 CNN for speech synthesis	69

5.4.5 Description of the machine learning process	71
5.5 The instrument	74
5.5.1 Admin's part	75
5.5.1.1 Adding book	76
5.5.1.2 List of all books	76
5.5.1.3 List of users	78
5.5.2 Client's part	79
5.6 Evaluation	82
5.7 Discussion	84
5.8 Summary	85
6 IMPLEMENTATION OF ASR SYSTEM FOR KAZAKH LANGUAGE	86
6.1 Methodology	86
6.1.1 Transfer learning on speech recognition	87
6.2 Datasets	90
6.3 Technologies	91
6.4 Experiment	91
6.5 Results and evaluation	92
6.6 Discussion	94
CONCLUSION	96
REFERENCES	97
APPENDIX A – Reading audio files	106
APPENDIX B – Standardizing audio files	107
APPENDIX C – Transformation of sequence	108
APPENDIX D – Reading all text files	109
APPENDIX E – Dictionary of the alphabet	110
APPENDIX F – Decoding the text	111
APPENDIX G – Restoring the weights	112
APPENDIX H – Dictionary of optimal characters	113
APPENDIX I – Extraction process	114
APPENDIX J – Griffin-Lim algorithm	115

NORMATIVE REFENRECES

In this dissertation, references are made in the following standards:

“Instructions for the preparation of a dissertation and author’s abstract” VAK
MON RK, 377-3

GOST 7.1-2003. Bibliographic record. Bibliographic description. General
requirements and compilation rules.

DEFINITIONS

Automatic speech recognition – is an ability of a computer to transform the human speech into text format.

Speech synthesis – is the ability of computer to transform the written text into audio speech.

Dataset – collection of data downloaded or collected for any machine learning tasks

Neural networks – is a network of structured neurons (units) that processed the incoming input data imitating the human brain.

Audio – data that represents the sound usually presented in binary format.

LIST OF ABBREVIATIONS

ASR - Automatic speech recognition

HMM - Hidden Markov Model

ANN – Artificial neural network

DNN – Dense neural network

CNN – Convolutional neural network

RNN – Recurrent neural network

LER – Label error rate

LSTM – Long short term memory

BiLSTM – Bidirectional long short term memory

1. INTRODUCTION

Automatic speech recognition (ASR) is a dynamic direction in the field of artificial intelligence. Over the past half century, a significant progress has been made in this area - there are many commercial applications that make investments in this area justified and profitable. Among such applications, first, it may be noted the introduction of call centers or IVR systems (Interactive Voice Response) - systems for automatic access to information, bypassing the operator. AT modern call centers questions are formulated by the user on natural language, and the answer is synthesized by the computer also in the language user. The introduction of call centers has freed up a huge number of operators and improved the service quality in many airports and train stations. Automatic speech recognition systems are widely used in medical research requiring input when hands the operator is busy (x-ray), or when you want to manage autonomous apparatus for the study of internal organs. Even filling out the medical cards by mid-level personnel in advanced medical facilities is done using voice. ASR systems have shown significant progress in modern voice assistants in every day used smartphones like Siri, Google Assistant, Alexa and many more. Other than speech-to-text algorithms they use natural language processing in order to analyze the sentence to provide best results for the smartphone user.

An important area of application of automatic recognition systems and speech synthesis is helping people with disabilities like problems with musculo-skeletal system and impairment of vision (assistive technology).

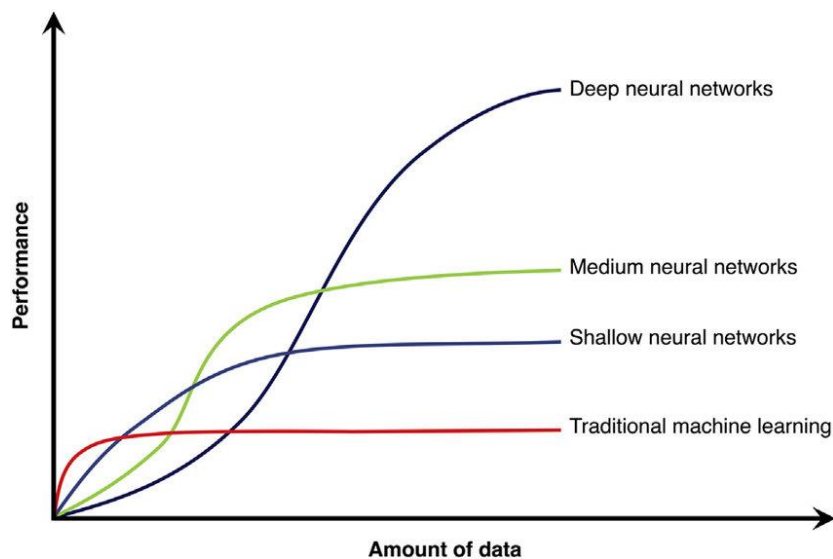


Figure 1.1 – Neural network performance over data graph

All these different areas that use automatic speech recognition require the big data in order to function accurately. Since the deep neural networks because of the recent leap forward in graphical processing unit (GPU) performs a lot better than medium neural networks (Figure 1.1) the number one problem that needs to be solved is a data problem.

1.1 Problem statement

It should be noted that automatic speech recognition systems in Kazakhstan almost never applied, which leads to an active researches and explorations of the area for developers. The main reason behind poor investigation and research in area of speech recognition for Kazakh language is poor amount of speech data. As it was observed in popular languages like English, Spanish and Chine, good quality ASR systems require tremendous amount of data. Popular speech corpuses like TIMIT or Switchboard contain massive amount of transcribed audio recordings with various types of speeches, like telephone speeches, conversational speech or clean microphone speeches. In Kazakh language there are almost no decent speech corpuses available on web-sources. The available ones are usually not free for usage and certainly are not enough to obtain powerful and effective ASR models. In order to build the decent speech corpus for ASR system, it requires a lot of time, well-structured environment and reliable monitoring system. However, to completely get rid of the issue related with the data deficit, the neural network structure and approach should be considered as well.

1.2 Aims and objectives

In order to construct the proper ASR system avoiding the problem of data deficit our main objectives are the following:

- To provide an well-designed environment for speech data collection
- To collect the significant amount of speech data with transcription for Kazakh language
- To apply post-processing technique on collected data and structure the files for neural network operation
- To apply the multilingual technique with Russian language by using transfer learning approach

1.3 Novelty

The novelties that were obtained during the execution of objectives are the followings:

- The application of a novel approach of speech data collection for any language by building reliable website with well-designed monitoring system and control
- The construction of an automatic post-processing step for speech data after collection process, that structures the speech and transcription with respect to training of CTC based neural network
- The training of neural network using multilingual approach by transferring the knowledge obtained from pre-trained Russian language model
- The automatic alignment of speech input sequence to transcription output sequence

1.4 Thesis outline

The thesis is structured in the following way: Chapter 2 overviews the speech signals processing. It considers all types of speech signals, noise removing algorithms and speech enhancement techniques like Kalman filtering, Adaptive Noise cancellation etc. Chapter 3 explores the analysis of building ASR system. It considers all existing and novel steps of building speech recognition system, reviewing the acoustic model, language model and the decoder. It also explains in detail the feature extraction step, which considered being very important step in all ASR systems. Chapter 4 analyses all novel end-to-end speech recognition systems like CTC based systems, RNN-transducer and Attention-based systems. It explores all advantages and disadvantages of each system over others explaining in detail their algorithms. Chapter 5 reviews the methodology built for collecting transcribed speech data. It explains step by step the technology itself and its benefits over other methods of data collection. In addition, it demonstrates the process of collection method in detail. Chapter 6 shows the process of building the ASR system for Kazakh language conducting an experiment. It explains deeply the novelty of the multilingual approach and transfer learning technology. Moreover, it runs through the whole experiment and visualization of results obtained by training the model. Chapter 7 concludes the whole research, marking the important part of the process.

2 SPEECH SIGNAL PROCESSING

Speech is the most obvious and natural way of communication between human beings. Throughout the conversation between humans there are being produced a lot of speech signals containing noises and external sounds. In building automatic speech recognition system (ASR) the speech signal processing is a crucial and important step. The performance of the ASR system is clearly dependent on high quality of the speech. However, the speech signal processing task is very difficult especially when it comes to removing the background noise from the speech [1]. In digital signal processing area the enhancement of the speech is showing a great impact on current speech recognition performances. With the involvement of the mathematical approach there are many speech enhancement methods. This section reviews the type of noises and algorithms and methods that enhance the speech by removing these background sounds etc.

2.1 Speech enhancement

Speech enhancement is a significant step in digital signal processing, which aims to increase the quality, provide the clarity of the speech using various algorithms and techniques. Figure 2.1 shows the basic illustration of the speech enhancement system [2]. There are different variations of the noises that lead to worthless speech signals.

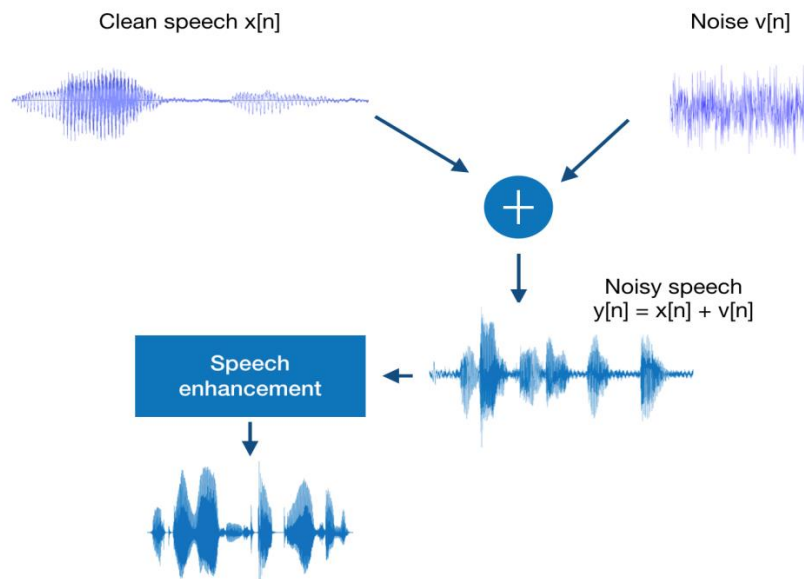


Figure 2.1 - Basic speech enhancement system

For speech, related applications like speaker verification, speech recognition, audio calls the clean and high quality speeches are strongly required. Speech enhancement methods can be different according to different types of noises.

2.2 Types of noises

This section describes the different types of noises and briefly explains their removal techniques. As mentioned above, for different types of noises and sounds the methods and techniques for speech enhancement can be different.

2.2.1 Wide band noise

In order to remove the wide band noise spectral subtraction and adaptive cancellation methods are used. In subtraction technique, calculated noise spectrum gets subtracted from the original noisy speech. Using adaptive cancellation the correlated noise is removed from the signal. The correlated signal can be achieved by estimated channel without a signal. Impulse of the adaptive filter should have the signal noise match the channel noise, so it can be removed later. The coefficients get updated once the output finds the minimum.

2.2.2 Removal of interfering speeches

Speech enhancement methods may struggle when two speeches interfere. If the pitches are identified the speakers can be differentiated from the signal. In order to separate the pitch voice segments of the speech should be tracked. To make the voice isolation of two speakers the domain transformation can be used as well. Knowing the pitch values of two speakers, we can calculate the Discrete Fourier Transform (DFT) of the signal and track the harmonics. If the DFT can be isolated we can separate the speaker voices [3].

2.2.3 Periodic noises

To remove the periodic noises stationary, adaptive filters and domain transformations can be used. In stationary approach, filters like twin T-filters are provided as a comb filter to remove the periodic noise. In the case of adaptive filter, forward prediction error approach is used as an inverse filter. In domain transformation noise removal can be achieved by differently manipulating the noise spectrum.

2.3 Speech enhancement techniques

There are a lot of speech enhancement methods. Basically, all these techniques can be divided into two categories: single channel speech enhancement and multi-channel speech enhancement.

2.3.1 Single channel enhancement

This technique doesn't provide second channel and commonly used in applications like mobile communication, hearing aid and so on. Compared to multi-channel this system is much easier and cost effective. Basically, this method uses the different speech statistics and noises [4].

2.3.2 Spectral subtraction technique

This method is one of the basic techniques in speech enhancement area. Spectral subtraction method assumes that speech signal is formed by addition of two components. Expression of this signal can be as follows:

$$y(t) = s(t) + d(t) \quad (2.1)$$

where time is t , $s(t)$ is a clean uncorrupted signal, $d(t)$ is the noise and $y(t)$ is damaged signal for further processing. The current signal gets cut into frames (overlapping) using window function and estimated in short-time Fourier transform domain and it is represented as follows:

$$Y(\omega) = S(\omega) + D(\omega) \quad (2.2)$$

The calculation of the power of noises can be handled as follows:

$$|Y(\omega)|^2 = |S(\omega)|^2 + \delta_n(\omega) \quad (2.3)$$

where $\delta_n(\omega)$ is the statistical average of $|D(\omega)|^2$ in non-speech time. Therefore, the following equations show the speech amplitude enhancement:

$$\begin{aligned} |\hat{S}(\omega)| &= [|Y(\omega)|^2 - E(|D(\omega)|^2)]^{1/2} \\ |\hat{S}(\omega)| &= [|Y(\omega)|^2 - \delta_n(\omega)]^{1/2} \end{aligned} \quad (2.4)$$

After the combination with the phase of noisy signal the representation will be the following:

$$S(\omega) = |\hat{S}(\omega)| e^{j\omega g[y(\omega)]} \quad (2.5)$$

To transform the signals back to time domain reverse short time Fourier transform is executed. However, the spectral subtraction estimation which assesses the durability during no speech period doesn't success on upgrading noise during speech stage.

2.3.3 Spectral subtraction with over subtraction model

To reduce the musical effect of the signal spectral subtraction with over subtraction model (SSOM) is used. SSOM makes the subtraction of an overestimated noise power spectrum and provides the resultant spectral components which are below the minimum spectral value.

2.3.4 Non-linear spectral subtraction

This technique is the combination of two proposals. First, are the extended noise usage and the model of over subtraction. Second, one is the nonlinear execution of a subtraction process, because this process depends on signal to noise ratio (SNR) frame [5].

2.4 Multi-channel enhancement methods

Multi-channel based systems are more complicated than single channel based systems. This method takes into account the inputs of a multiple signal to the system, and then with the help of adaptive noise cancellation uses the noise reference. These kind of systems work more effective on non-stationary noises than single channel based techniques because they take into consideration the spatial properties of the noises and signals [6].

2.4.1 Adaptive noise cancellation (ANC)

This technique is considered to be one of the effective and powerful enhancement methods. It is based on additional channels (reference path), where correlated signal of corrupted noise is happened. During the algorithm, the reference input is filtered and after this process the output gets subtracted from the original signal in which the noisy speech is present. The ANC neutralizes the noise by using so called reference signal. Reference signal cancels the anti-noise with the same amplitude and with an opposite phase. These signals are generated from several sensors which are placed near to the noise and interfering sources [7].

2.4.2 Beamforming

Beamforming technique is consists of advanced multidimensional filters, that increases the quality of wanted speech and cancels the unwanted noises. In this process, several microphones are places in a specific geometrical order. The beamformer then is used to

filter out the outputs and increases the signals based on their direction of arrival (DOA). The main purpose of this technique is based on assumption that there is a small amount of reflexions and the desired signal's DOA is already known. Based on the right alignment of the phases in sensors, the enhancement of the signal will be handled by cancelling the noisy parts that are not aligned in phase.

2.4.3 Wiener filtering

Wiener filtering technique in speech enhancement was proposed as an improved version of spectral subtraction. This technique is often used in a lot of speech enhancement techniques. The core idea behind Wiener filter is obtaining the estimation of a clean signal from unwanted using additive noise. It is achieved by minimizing the mean square error (MSE) between estimated signal and desired signal.

Filter transfer function in frequency domain gives us the following optimization equation:

$$H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_v(\omega)} \quad (2.6)$$

where, $P_s(\omega)$ is the power spectral density of the clear signal and $P_v(\omega)$ is the density of the noise. Considering that clean signal and noise signal as uncorrelated signal, the above equation can be derived. The SNR will be described as the following:

$$SNR = \frac{P_s(\omega)}{\hat{P}_s(\omega)} \quad (2.7)$$

The integration of the above formula with Wiener filter equation will be described as follows:

$$H(\omega) = [1 + \frac{1}{SNR}]^{-1} \quad (2.8)$$

The main disadvantage of the Wiener filtering technique is the estimation of the power spectral density of the signals before filtering and fixed frequency response.

2.4.4 Kalman filtering

The generalized version of Wiener filtering method is the Kalman filtering. This method contains varying autoregressive model (AR). Inside the framework of Kalman filter excitation model and AR model fit perfectly. They benefit from the capability of the

Kalman filter for handling non-stationary signals. The parameters of AR model are calculated using the help of linear predictive coding (LPD) analysis. For estimation of quickly time-changing excitation model with noise Multi-Pulse linear predictive coding (MPLPC) is used. Therefore, Kalman filter consisting all measured data, measurement devices and system knowledge provides the estimation of the parameters and variable, so that they give minimized error. The main difference between Kalman filter and Wiener filter is the ability of getting along with non-stationary signals [8].

2.4.5 Linear predictive coding

Linear predictive coding (LPC) most of the time is used to process the audio signal and speech signal processing for representing the speech digital signal in compressed manner. LPC make the assumption that sound starts at the end of the tube producing buzz sounds sometimes adding popping sounds. The buzz sound is produced by glottis, because it is well-known for its loudness and pitch. The tube gets generated by the vocal tract, which is known for its resonances (formants). The hisses and popping sounds are produced by the throat, tongue and lips.

LPC analyses the speech signal by using the so called formants, cancelling the effect from the speech signal and calculating the remained signals frequency and intensity. This process of removing these effects is called inverse filtering. The signal remained after the removal process is called residue. The frequency and intensity of the buzz, residue and formants can be transferred and saved. The core issue with LPC method is defining the formants from the initial signal. The solution for this problem is to represent the current sample as the combination of previous ones. The definition of the formants will be coefficients from the equation. In order to calculate these coefficients LPC system should be used.

2.4.6 DFT based method

This method is known for its simplicity and less complex computations. It uses the short time discrete Fourier transform (STDFT) and popular for its spectral processing technique. For the perception of the human speech these techniques are not responsive. Therefore, clean extraction of the spectral amplitude from the noisy signal must be done in order to have a suitable and good quality speech output. Hence, these methods are based on short time spectral amplitude.

2.4.7 Signal subspace method

This technique uses a signal dependent transform in order to decompose the noises into two different subspaces: first one is the noise plus signal subspace and second one is the noisy signal only subspace. This transformation operation is called karhunen loeve

transform (KLT). It makes an assumption that speech can exceed the signal with respect to noise subspace and noise only subspace. The elements of KLT that describe the noise-only subspace are cancelled, while the components that represent the noisy signal are changed with gain function. The improved signal is derived from inverse KLT of changed elements. The main purpose is to enhance the quality and minimizing the error of clarity. The improved speech is obtained by the signal subspace using the algorithm of adaptive noise estimation. However, this algorithm suffers the poor performance and updates the noise estimates in the absence of speech. Figure 2.2 shows the block diagram of the subspace speech improvement system.

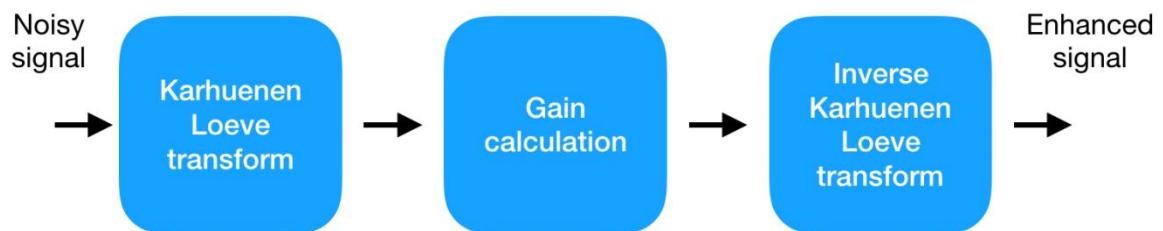


Figure 2.2 - Subspace speech enhancement diagram

3 ANALYSIS OF EXISTING ASR SYSTEMS

This section explores and discusses the components of traditional ASR system. Moreover, it analyses the disadvantages over the advanced methods of building ASR systems. Each component is described separately by exploring their algorithms and methods

3.1 Hidden Markov Model based architecture

The main principals of continuous speech recognition process with huge dataset consist of important components (Figure 3.1). The audio wave that was generated from a microphone passes through the feature extraction step that converts it to sequence of acoustical features vectors $Y = y_1, \dots, y_T$. The job of the decoder is to find the word sequence w that maximizes the following probability:

$$\hat{w} = \arg \max \{P(w/Y)\} \quad (3.1)$$

By using the Bayes' Rule the probability $P(w/Y)$ can be transformed to the following equation:

$$\hat{w} = \arg \max \{p(Y|w)P(w)\} \quad (3.2)$$

The probability $p = (Y/w)$ is defined by an *acoustic model* which learns the relationship between audio sound and phonemes.

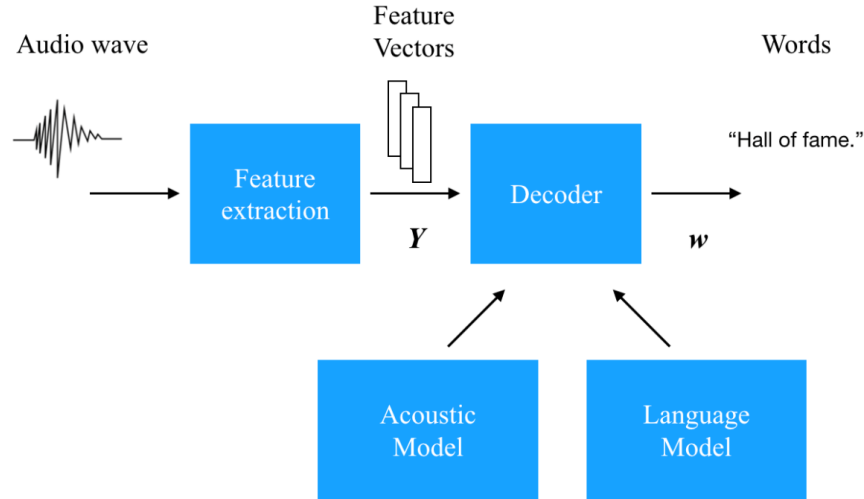


Figure 3.1 - Automatic speech recognition architecture

The probability $P(w)$ is defined by a *language model* that represents the probability of each word in N -gram model based on the previous $N-1$ words.

The acoustic model for given w is combined with phone models in order to make words, which are trained using a dataset of speeches with its corresponding orthographical transcriptions. The language model is a probability distribution of word sequences which is trained on large text corpora.

3.2 Feature extraction

Feature extraction step provides the numerical representation of the speech signal. This representation should provide the minimal information loss compared to the original speech. It can be interpreted as dimensionality reduction technique that transforms the original full-size data into reduced-size data without influencing ending outcome.

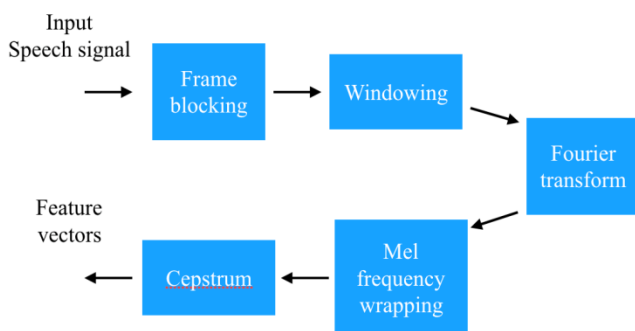


Figure 3.2 - MFCC extraction steps

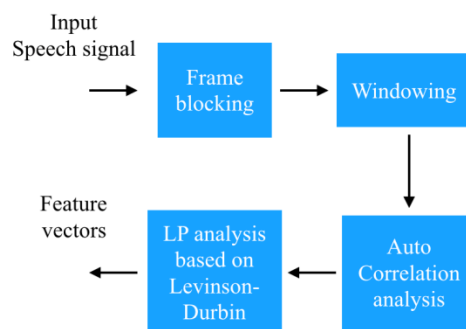


Figure 3.3 - LPC extraction steps

Feature extraction widely used and applied in a lot of tasks like speaker verification, gender recognition, language recognition [9, 10, 11, 12, 13, 14, 15] etc.

For automatic speech recognition tasks there are 2 popular feature vector representations which are *linear predictive coding*[16, 17] and *Mel-frequency cepstrum coefficient*[18, 19].

3.2.1 Linear predictive coding (LPC)

The main idea behind LPC is the approximation of a sample by linear combination of previously constructed samples. The LPC extraction process components can be observed in the Figure 3.3.

In order to calculate the LPC features, speech signal firstly gets blocked into frames of N samples. Secondly, after frame blocking process is finished, each separated frame goes through a windowing process to minimize the breaks at the start and the end of the frame. In the next process, each windowed signal frames will auto correlate with each other to

go to the LPC analysis step. LPC analysis, by using Levinson Dubrin's algorithm each auto correlation will be converted into LPC parameter set.

Based on the linear prediction operation, if the x_i is the current audio signal, the predicted value \hat{x}_i based on the previous p samples can be calculated by the below equation:

$$\hat{x}_i = -a_2x_{i-1} - a_3x_{i-2} - \dots - a_{p+1}x_{i-p} \quad (3.3)$$

Where $1, a_2, \dots, a_{p+1}$ are the filter coefficients. With a given signal LPC filter produces the feature vectors which describe the variance of the predicted signal.

3.2.2 Mel-frequency cepstrum coefficient (MFCC)

MFCC are one of the most popular and often used feature vectors in Automatic speech recognition (ASR) system [19, 20]. The MFCC feature vectors extraction process is illustrated in Figure 3.2.

The MFCC try to imitate the behavior of the human ear's bandwidths having frequencies below 1 kHz. It is calculated by dividing the audio signal into overlapping frames. If the frames are divided into N samples and for each frame hamming window is multiplied and the equation is:

$$W(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad (3.4)$$

In the next step, signal will be converted from time domain into frequency domain by using Fourier transform. The equation of Discrete Fourier transform (DFT) for the signal is the following:

$$X_k = \sum_{i=0}^{N-1} x_i e^{-\frac{j2\pi ki}{N-1}} \quad (3.5)$$

After DFT calculation step, the Mel for a defined frequency is calculated by converting the frequency domain into Mel frequency scale, which is suitable for human perception. This is handled by triangular filters in order to make the approximation of a Mel scale. The equation for calculating the Mel scale with a given frequency is the following:

$$M = 2595 \log_{10}\left(1 + \frac{f}{700}\right) \quad (3.6)$$

The next step converts the Mel scale into time domain using discrete cosine transform (DCT). The calculation of DCT is done by the equation below:

$$X_k = \alpha \sum_{i=0}^{N-1} x_i \cos\left\{\frac{(2i+1)\pi k}{2N}\right\} \quad (3.7)$$

After the step above, we end up with acoustic vectors which are called Mel frequency Cepstrum Coefficient. The MFCC feature vectors are good for representing the individual characters of each speech for further recognition tasks like speaker identification.

3.3 Acoustic modeling using HMM

Acoustic model is for creating the statistical probability of speech features for each phone unit. Figure 1 shows that the acoustic model is used to learn and analyze the speech features for their acoustical characteristics. In recognition process, the unseen words are recognized by comparing it with already seen templates and finding the closest one. But these templates cannot match the acoustical changes. Therefore, acoustical models are built by probability distribution over the acoustics. The most basic way is by using Gaussian distribution which finds the representation parameters [21].

Speech recognition task initially considered as a statistical classification task. Classes defined as words sequence W from the large vocabulary set and input defined as the features of the speech X . The equation looks like as following:

$$P(W|X) = \frac{P(X|W)P(W)}{P(X)} \quad (3.8)$$

The ASR system with HMM is used to model a word in a vocabulary, in which every hidden parts represents the phoneme units, whereas observable parts takes in to account the characteristics of the acoustic events. The samples of the events are being collected and used for training algorithms like Baum Welch or Viterbi for learning the right evaluation of HMM parameters. After the training is finished, the recognition part of the HMM can be used. The recognition means for every new sequence there will be assigned the most likely model using maximum likelihood (ML) criteria. Despite that, HMMs have limitations. Since the standard HMMs based on Baum Welch or Viterbi training algorithms rely on ML criteria, they show poor distinguishing power compared to different models. The traditional HMMs are based on strong assumptions of properties of the current event.

Later the HMM/GMM speech recognition model was constructed, where HMM modeled the sequential structure of the speech and GMM model served as an acoustical

characteristics for each HMM state. A GMM's weighted M Gaussian densities defined as follows:

$$P(x|\lambda) = \sum_{i=1}^M w_i g(x|\mu_i \Sigma_i) \quad (3.9)$$

where, $p(x|\lambda)$ is the probability of D dimensional feature vector with parameters $\lambda = \{w_i, \mu_i, \Sigma_i\}$ where w_i are the weight which meets $\sum_{i=1}^M w_i = 1$. The component Gaussian densities are the following:

$$g(x|\mu_i, \Sigma_i), i = 1, 2, \dots, M \quad (3.10)$$

where each density is a D-variate Gaussian function:

$$g(x|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \right\} \quad (3.11)$$

where μ_i is a mean vector and Σ_i is a covariance matrix [22].

GMM treats the signals as if they consist of different components which are independent from each other. The components consist of acoustical classes that define the vocal tract configurations [23]. Therefore, it is tend to model the features as if the words having their individual characteristics. Each component is optimized using the algorithm called EM [24]. This algorithm is used to calculate the maximum probability distribution from not full data like missing feature vectors. One of the simplest way of combining these models (HMM and GMM) is to 3 state HMM and set the GMM in the middle state to get the observation probabilities for the current state [25].

Although, HMM/GMM techniques have been popular and used a lot for speech recognition tasks, they have several limitations. Firstly, they use the MFCC and PLP for feature extraction methods. Since these features are short time spectral, they are not able to capture the whole long time dependency in which phones are spread. Secondly, in HMM/GMM there is an assumption where each observation frame is conditionally independent in given the state sequence. This means that observations depend on the previous observation, not on neighbors. Since the ASR is pattern recognition (classification) problem, the machine learning techniques like artificial neural network (ANN) [26], support vector machine (SVM) have been proposed as an appropriate solution to acoustic modeling problem.

3.4 Acoustic modeling using ANN

One of the main advantages of ANN over other modeling methods in speech recognition it can approximate the nonlinear dynamic systems. Speech is a nonlinear signal made by nonlinear system.

ANN is basically an interconnection of computational elements (neurons) and this nonlinear system is distributed through the network. The basic nonlinear neuron model is shown in Figure 3.4. In this figure neuron k receives the inputs x_1, x_2, \dots, x_N from previous N neurons, which have a connection weights w_1, w_2, \dots, w_N or $w_{k1}, w_{k2}, \dots, w_{kN}$. Then the output neuron will be described as follows:

$$y_k = f(u_k + b_k), u_k = \sum_{j=1}^N x_j w_{kj} \quad (3.12)$$

where b_k is an internal offset and f is a nonlinear function.

Since the several simple nonlinear computational elements can construct the large neural network, it can approximate any nonlinearity. Therefore, neural networks provide a more effective way of modeling nonlinear transformation between input and output. Thus approach (connectionist models) with nonlinearly interconnected elements can be fully applied to speech modeling.

At the early stage, usage of ANN for speech recognition was successfully attempted on recognizing simple digits, few phonemes and words [27] using multilayer perceptron. Although it was basically used for acoustic modeling, this method had a significant limitation on capturing the temporal information of the speech signal. In order to solve this issue, recurrent neural network (RNN) and time delay neural network (TDNN) were suggested.

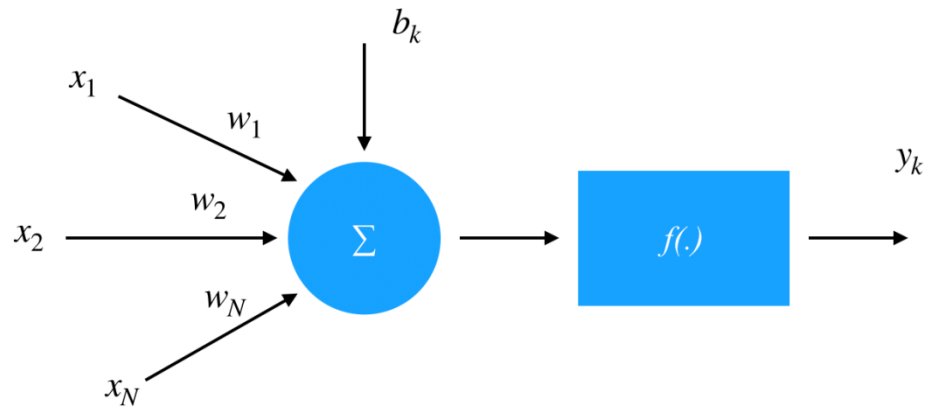


Figure 3.4 - Nonlinear neural model

TDNN is an expansion of multilayer perceptron for time-sequence data. TDNN turns the temporal sequence data into spatial data for each unit and can be back-propagated to training. But on the other hand, RNN due to its neural network architecture has the ability to capture the dynamics of temporal sequence. In [28], the description of phoneme recognition using TDNN is presented. The hierarchy using 3 layers of simple computing neurons is constructed to form the arbitrary nonlinearity. TDNN learns this using simple back-propagation algorithm. The time delay process allows the network to learn the phonetic features and relationship between them. In [29] paper work, authors present the word recognition based system on completely recurrent neural network, which is trained by back-propagation in time. Recognition process is done by using dynamic programming, which tends to find the most possible set of symbols. Works [30, 31] also mention the effectiveness of RNN for word and phoneme recognition. In [32] paperwork, authors describe the ability of RNN to model the phone probability estimation on large vocabulary speech. They have concluded that, RNN is entirely effective to invoke the context information.

Although ANN based models were popular, they never succeeded on working with long time sequences. That's why; researchers targeted their attention on using HMM for temporal modeling and ANN for phonetic modeling. This hybrid model improves the flexibility of HMM/ANN and overall recognition process. The main reason is that ANN based models are more capable than GMM to capture the dynamic information of the features in extended windows. Other than that feed forward ANN while estimating the posterior probability, does not need to have detailed assumptions of data.

In HMM/ANN based systems, ANN is tend to output the posterior probability for each state. The probability outputs will be converted to scaled likelihood and described as follows:

$$\frac{p(x_t|q_k)}{p(x_t)} = \frac{p(q_k|x_t)}{p(q_k)} \quad (3.13)$$

x_t is a feature vector in time t , q_t is a k -th state of HMM states [33]. Presented scaled likelihood is used to replace the HMM/GMM's state likelihood that was modeled in GMM. In early stages, ANN used MLP with one hidden layer and output layer with one unit for each class. It was trained on labeled data in order to maximize the mutual context between input features and target outputs. The size of ANN whenever HMM/ANN used the sub-word units' phonemes is usually one ANN output for each phoneme. It uses the scaled likelihood from outputs for all state of the phoneme.

There were a lot of ways to describe and present the combination of HMM (time sequence modeling) and ANN (model the speech signal). The authors in [34] have replaces GMM with MLP as a probability estimator in HMM based model. They have

showed the work where they model the tri-phone units with MLP and proved that it leads to a better performance and accuracy compared to GMM. Recently, the authors in [35] have presented a new RNN architecture approach called Long-Short Term Memory (LSTM) as shown in Figure 3.5.

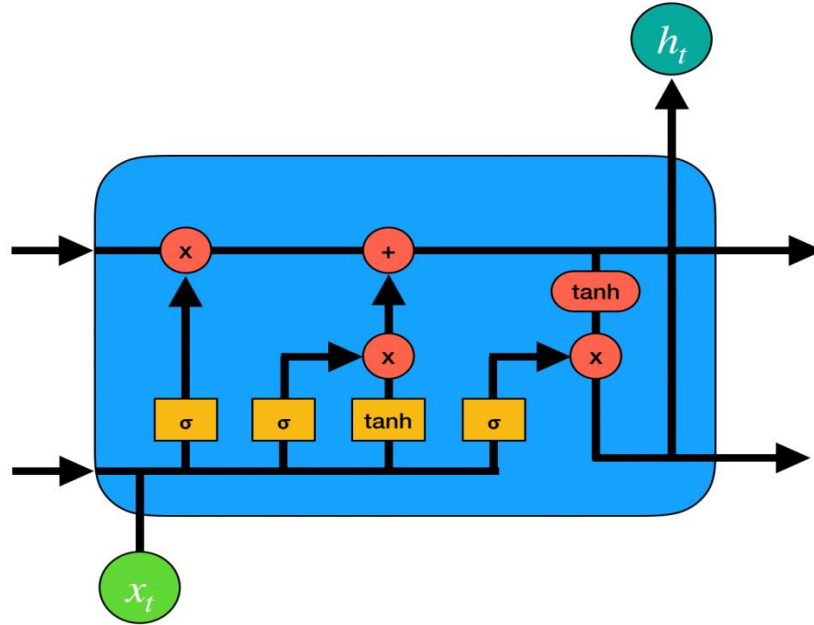


Figure 3.5 - LSTM architecture

This architecture was introduced to address the exploding and vanishing gradients in standard RNN. Compared to simple feedforward ANN, recurrent networks have cycling connections which make them more effective in sequential data. Therefore, they have shown that LSTM based recurrent models have more power over solving large vocabulary speech recognition tasks. Authors in [36] have also explained the core advantages of RNN based models on phone sequence modeling.

In recent years, the researches that work with HMM/ANN hybrid system are paying attention on deep belief neural networks (DBNN) for modeling acoustics. In the following section, we will do a review on DBNN acoustic modeling.

3.5 Acoustic modeling using DBNN

Deep belief neural network (DBNN) has been discovered as powerful and efficient for many machine learning problems as well as for acoustic modeling in HMM/ANN based speech recognition system. At first, DBNN was presented for acoustic modeling tasks, because it had much higher capacity for modeling than regular GMM. Moreover, DBNN also showed very effective training ability that merges the unsupervised learning for feature discovery and supervised learning for optimization and fine-tuning of the features.

Another reason that DBNN is efficient is that the low-level feature characteristics are computed in lower layers and highly nonlinear structure of the input are taken care of in higher layers. It can be similar to human speech recognition that uses a lot of layers for features extractions [37, 38].

DBNN consist of many hidden nonlinear units and very huge output layer. HMM takes the phones on either side and each phone can be presented by a number of tri-phones. Since the states of these tri-phones are bonded together, HMM produces thousands of these states. Therefore, DBNN's output layer can be very effective to handle large number of states. Nowadays, due to rise of hardware capacity and machine learning algorithm DBNN can be trained easily. Recent researches that use new learning technologies have shown that DBNN outperforms the traditional GMM at acoustic modeling task in speech recognition on various data's.

DBNN is a graphical model that has many layers of binary variables that is learned effectively one layer by one layer using unsupervised learning process. The way of its learning allows using the forward generative weights in the reverse direction. This gives us the ability to treat this graphical model as a simple ANN, which will be fine-tuned and optimized using back-propagation. During the training process, every hidden neuron j usually uses the logistic function to cover the whole input from the previous layer x_j . It maps it to scalar state y_j and sends it to the next layer. The logistic function is the following:

$$y_j = \text{logistic}(x_j) = \frac{1}{1+e^{-x_j}} \text{ and } x_j = b_j + \sum_i y_i w_{ij} \quad (3.14)$$

b_i is the bias of j unit, i is an index over neurons in previous layer. w_{ij} is the connection weight between unit j and i in the previous layer. In the case of multiclass classification, the input x_j from unit j is converted to class probability p_j using softmax:

$$p_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)} \quad (3.15)$$

k in an index over all classes [39].

DNN can be trained using back-propagation by taking derivatives of the cost function which measures the difference between the target and the true output. In softmax output function, the cost function C is the cross-entropy between target d and softmax outputs p .

$$C = - \sum_j d_i \log p_j \quad (3.16)$$

where target probability holds the value one or zero and it is provided to train the DBNN classifier model. It is more efficient to train the large dataset by computing the derivatives on small parts of the data (mini-batch) rather than whole training data. The weights get updated each time one mini-batch is completed. Further, the stochastic gradient descent can be applied by using the momentum value α that varies from 0 to 1, which smooths out the mini-batch t gradient.

$$\Delta w_{ij}(t) = \alpha \Delta w_{ij}(t-1) - \epsilon \frac{\partial \mathcal{C}}{\partial w_{ij}(t)} \quad (3.17)$$

Updating process for biases can be achieved same as weights on connections, which are coming from units.

There are a lot of works on acoustic modeling using DBNN. The phone recognition process on TIMIT dataset described by the authors in [39] was achieved by replacing the GMM with DBNN. In this experiment, network first was pre-trained in order to create the multilayer generative model for features with no usage of discriminative information. After providing these information models are adjusted to predict the probability distribution over HMM states. It was trained with back-propagation. Authors have considered the 2 types of DBNN, back-propagation DBNN and associative memory DBNN. Both networks have techniques to prevent over fitting problem. The prevention of over fitting in back propagation DBNN is accomplished by using the bottleneck in the last layers, while hybrid generative training is used to prevent it from associative memory DBNN. Both architectures, completely outperforms the other architectures with variety of hidden layer numbers and numbers of units per layer. The training is done on TIMIT dataset and the best result that DBNN has shown 23% of phone error rate (PER).

In the work [40] authors have used the DBNN for context dependent model on large vocabulary speech. DBNN/HMM hybrid system which was described in this work is trained the DNN to output the distribution over tri-phones states. During the whole experiment, it was illustrated that context dependent DBNN/HMM system can notable outperform the context dependent GMM/HMM with a significant improvement of 5.8% and 9.2% (minimum phone rate and maximum likelihood, respectively).

Authors in [41] have shown that the word recognition accuracy can be improved using DBNN by structuring it in a hierarchical way to model the long-term energy paths. It has been tested on 5000 word Wall street journal task and showed the noticeable enhancements on phone and word recognition.

In this [42] work the modular combination of 2 well known ANNs is proposed for large continuous speech recognition. DBNN firstly is trained to extract the bottleneck features from mel scale filter bank coefficients. Then the network is used as nonlinear discriminative feature for hybrid system. The acoustic modeling part is done by DBNN. This effectively influenced the very large network. Authors have concluded that

bottleneck features significantly enhance the recognition process of DBNN/HMM. The modular combination makes the acoustic model work well with large context. The evaluation is done on Tagalog data consisting telephone speech.

The evaluation of phoneme class conditional probabilities is described using convolutional neural networks in [43]. The authors proposed the approach on TIMIT phone recognition, which consist of ANN inputting the raw speech signal and outputting the phoneme class conditional probability. CNN based architecture showed much significant leap forward compared to regular MLP architecture. Authors have concluded that CNN is much suitable for temporal signals. With no pre-processing step CNN based model slightly outperformed the baseline. The results state that deep structured networks can learn much important features.

Graves, Mohamed and Hinton in their work [44] propose the deep recurrent neural network for speech recognition. The end-to-end system called connectionist temporal classifier (CTC) has trained the RNN for sequence to sequence tasks where there is no alignment between input and output. On TIMIT phoneme dataset with all necessary regulation techniques they have trained the long short term Memory (LSTM) based RNN. As a result, they have achieved the test set error 17.17%, which was a very impressive outcome.

In [45] authors investigated the reverberant speech recognition problem using DBNN. Their system contains auto encoder for improving the speech features and acoustic model based on DNN/HMM for recognition. They have estimated their result on the task presented in Chime Challenge 2014. DNN/HMM trained on multi-condition data has achieved higher accuracy compared to GMM/HMM system trained on the same dataset. Moreover, the auto encoder for feature improvement contributed to overall recognition performance as well.

Work that was reported here [46] presented the distributed DBNN for multilingual acoustic modeling. Authors showed the comparison of cross, mono and multilingual acoustic model using deep networks and the same with baseline models on 10 000 hours of Romance languages. The average improvements over baselines are 4%, 7% and 2% for mono, cross and multilingual respectively.

3.6 Language models in ASR

In general, good designed and well-structured language model (LM) makes a huge impact not only on speech recognition [37, 44], but machine translation [47, 48], semantic extraction [49] and etc. For this reason, Language modeling has been popular and most of the researchers have been focusing in this area, publishing papers in recent years. The state of the art approach in Language modeling has been N-gram Language model [50]. However, the neural network based LM has gained a lot of attention due to its performance and potentials over other LM methods.

There has been an attempt in early stages to introduce the ANN into Language modeling tasks in [51, 52]. However, the neural network language models (NNLM) started to attract researchers only after recurrent neural networks have been applied [53, 54]. Only after constant research, significant improvements NNLM started to be considered as the technique which can overcome the state of the art.

3.6.1 N-gram Language model

The probability of a word sequence in (1.2) is given by:

$$P(w) = \prod_{k=1}^K P(w_k | w_{k-1}, w_{k-2}, \dots, w_1) \quad (3.18)$$

For recognition of huge vocabulary, the above equation (3.18) can be shortened to form an N-gram language model

$$P(w) = \prod_{k=1}^K P(w_k | w_{k-1}, w_{k-2}, \dots, w_{k-N+1}) \quad (3.19)$$

The N-gram language model likelihood is evaluated from text data by counting occurrences to represent the maximum probability. If $C(w_{k-2}w_{k-1}w_k)$ presents the occurrences of the words $w_{k-2}w_{k-1}w_k$ and $C(w_{k-2}w_{k-1})$ represents the similarity between two words, then:

$$P(w_k | w_{k-1}w_{k-2}) \approx \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} \quad (3.20)$$

The problem that occurs with the likelihood equation above (3.20) is the data sparsity. This is handled by so called discounting using *Katz smoothing* [55] with the following equation:

$$P(w_k | w_{k-1}w_{k-2}) = \begin{cases} d \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} & \text{if } 0 < C \leq \hat{C} \\ \frac{C(w_{k-2}w_{k-1}w_k)}{C(w_{k-2}w_{k-1})} & \text{if } C > \hat{C} \\ \alpha(w_{k-2}, w_{k-1})P(w_k | w_{k-1}) & \text{otherwise,} \end{cases}$$

(3.21)

where \hat{C} is a counting threshold, d is a discount coefficient, α is a normalization. If the N-gram count exceed some threshold maximum likelihood calculation is used. If the count is small it still uses the maximum likelihood estimate, but with a slight reduction. The discount coefficient calculated on the basis of Turing Good formula $d = (r + 1)n_{r+1}/rn_r$ where n_r N-grams quantity that appear r exactly r times in training set.

In order to build more robust and effective language models class based models [56] can be used, where every word w_k has a corresponding class c_k :

$$P(w) = \prod_{k=1}^K P(w_k|c_k)p(c_k|c_{k-1}, \dots, c_{k-N+1}) \quad (3.22)$$

Although, the class based N-gram models calculated using maximum likelihood the data sparsity problem is very rare since the classes are noticeably less than words. It is useful for making the training set likelihood as optimized as possible.

The robust language model for large vocabulary with standard training sets (10^7 words) usually consist of 3-4 gram words based models interpolated with class based models.

3.6.2 Basic neural network based LM

The aim of the statistical LM is to calculate the probability of the word sequence $w_1 w_2 \dots w_T$ in the language and the equation can be presented multiplication of conditional probabilities of each word based on the previous ones:

$$P(w_1^T) = \prod_{t=1}^T P(w_t|w_1^{t-1}) \quad (3.23)$$

where, $w_i^j = w_i w_{i+1} \dots w_{j-1} w_j$. This rule based on the assumption where the particular words in a word sequence dependent only on previous context. NNLM can be considered as statistical model and as a neural probabilistic model. In term of the architecture variation of ANN, NNLM can be separated as follows: Feed forward NNLM, recurrent neural network language model (RNNLM) and LSTM based RNNLM.

3.6.3 Feed forward neural network language model (FNNLM)

The goal of the FNNLM is to assess the conditional probability $P(w_t|w_1^{t-1})$. However feed forward neural network is not capable of representing the history context. Therefore the N-gram based model has been applied to FNNLM, where words in word sequence dependent on the words located near to them. Considering only $n-1$ direct previous words the evaluation of the conditional probability is the following:

$$P(w_t|w_1^T) \approx P(w_t|w_{t-n+1}^{t-1}) \quad (3.24)$$

The original FNNLM architecture is illustrated in Figure 3.6, where w_0 and w_T are the beginning and ending places of words sequence.

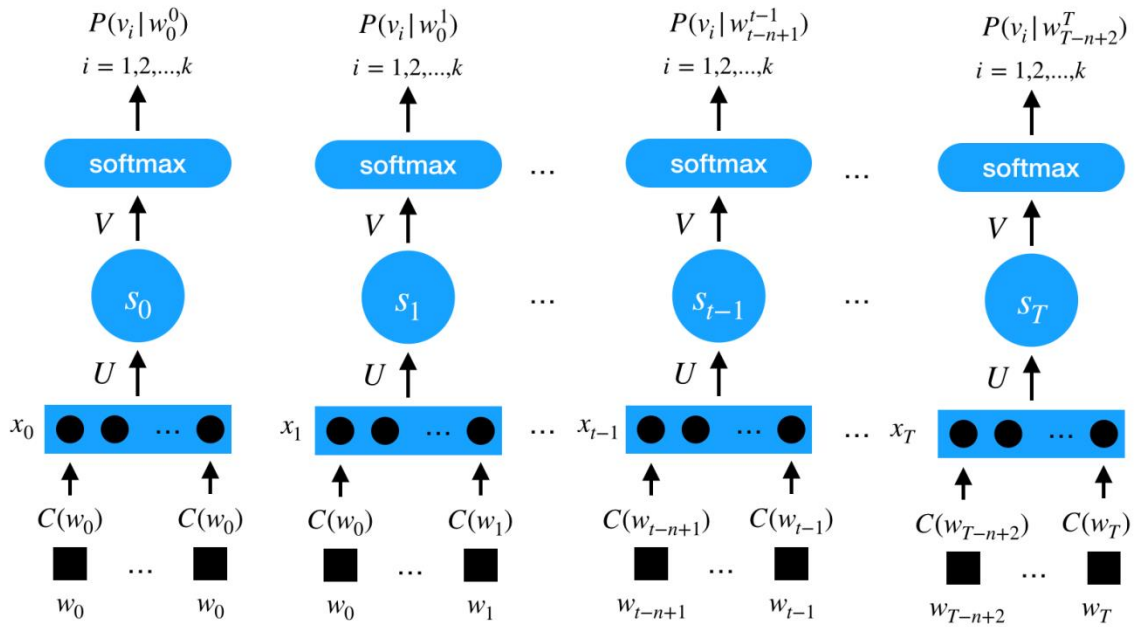


Figure 3.6 - Feed forward neural network language model architecture

This model has the vocabulary which contains words with their corresponding unique index. To calculate the conditional probability of the particular w_t , its previous $n-1$ words $w_{t-n+1}, \dots, w_{t-1}$ are converted into feature vectors by shared matrix $C \in R^{k \times m}$ according to their unique index in the vocabulary (m is the feature vector dimension and k is the size of the vocabulary). Every row in the projection matrix C corresponds to one word in the vocabulary. The input $x \in R^{n_i}$ of feed forward network is obtained by combining feature vectors $w_{t-n+1}, \dots, w_{t-1}$ in which the size of the layer is $n_i = m \times (n - 1)$. So, the representation of the FNN can be as follows:

$$y = V * f(U * x + b) + M * x + d \quad (3.25)$$

where $U \in R^{n_h \times n_i}$, $V \in R^{n_o \times n_h}$ are the weight matrixes, n_h is the number of hidden layer, n_o is the output layers size, direct connection between input and output layer is defined by the weight matrix $M \in R^{n_o \times n_i}$, biases in hidden and output layers defined by b and d , $y \in R^{n_o}$ is an output vector and the activation function is described with $f(.)$.

To be certain that all words conditional probabilities are positive and sum up to one, a softmax layer describes the output layer as follows:

$$P(v_i|w_1^{t-1}) \approx P(v_i|w_{t-n+1}^{t-1}) = \frac{e^{y(v_i, w_{t-n+1}^{t-1})}}{\sum_{j=1}^k e^{y(v_j, w_{t-n+1}^{t-1})}}, i = 1, 2, \dots k \quad (3.26)$$

where v_i is the i -th element in the vocabulary and $y(v_i, w_{t-n+1}^{t-1})$ is the i -th element of the output vector.

Training of the network is done by maximizing the log-likelihood of the given data as described below:

$$L = \frac{1}{T} \sum_{t=1}^T \log(P(w_t|w_1^{t-1}; \theta)) + R(\theta) \quad (3.27)$$

In which θ is parameters of the model for training and $R(\theta)$ is regularization.

The most commonly used learning algorithms for training the neural network is the stochastic gradient descent (SGD) with back-propagation. The recommended selection for loss function is the cross-entropy. The parameters get updated as follows:

$$\theta = \theta + \alpha \frac{\sigma L}{\sigma \theta} - \beta \theta \quad (3.28)$$

where α and β are learning rate and regularization parameter, respectively.

The overall performance of the NNLM is defined using perplexity (PPL) which is described as follows:

$$PPL = \sqrt[T]{\prod_{i=1}^T \frac{1}{P(w_i|w_1^{i-1})}} = 2^{-\frac{1}{T} \sum_{i=1}^T \log_2 P(w_i|w_1^{i-1})} \quad (3.29)$$

3.6.4 Recurrent neural network language model (RNNLM)

RNN based neural networks are completely different from usual feed forward neural networks. Other than working only on input space RNN can operate on internal state space, which allows capturing and enabling the sequentially long-time dependencies. Therefore, any length can be handled by RNNLM, because the prediction of the next word based on memorizing all previous contexts. As illustrated in Figure 3.7, the structure of words in RNNLM is quite similar as in FNNLM. However, the input to RNN at every time step t is the feature vector of the previous word, not the combination of all previous words ($n-1$).

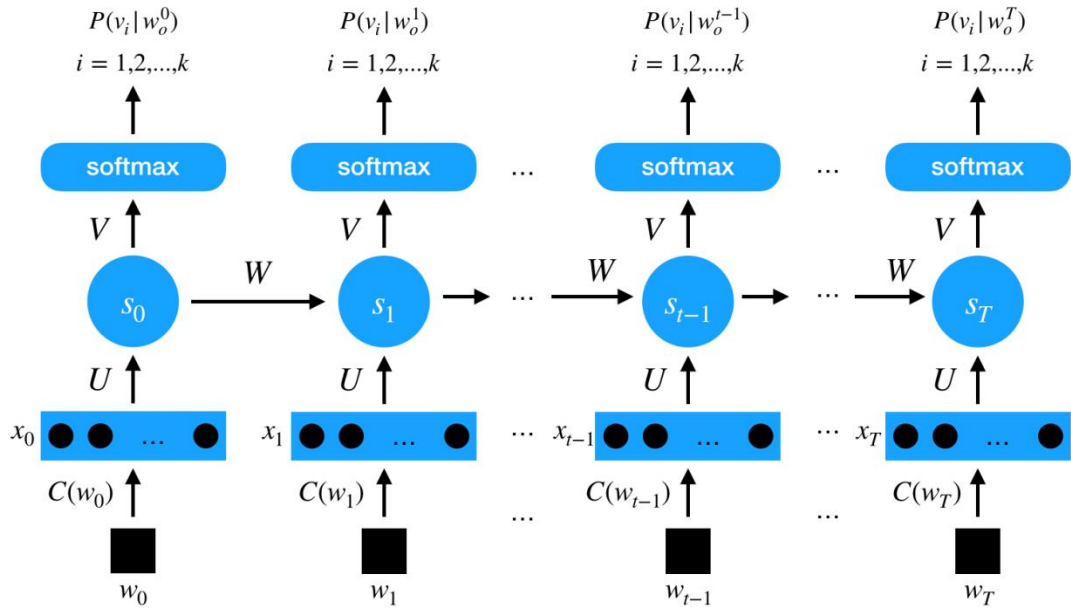


Figure 3.7 - Recurrent neural network architecture

The concatenation of previous words are saved and counted using the internal state of the previous step. At each time steps, the description of the RNN as follows:

$$\begin{aligned} s_t &= f(U * x_t + W * s_{t-1} + b), \\ y_t &= V * s_t + M * x_t + d \end{aligned} \quad (3.30)$$

where, $W \in R^{n_h \times n_h}$ is the weight matrix, $n_i = m$ is the size of input layer. The RNN's output layer should be normalized and regularized by softmax layer.

The back-propagation through time (BPTT) algorithms is used for training part because of the presence of internal state of the previous steps at each time. If the training data is happened to be single and long word, the short version of BPTT is used. For small corpuses, usually 5 steps are enough for error calculation using back-propagation.

3.6.5 Long Short Term Memory RNN Language model

Even RNNLM has the ability of taking into consideration of the previous words; it is very hard to train the dataset with long time dependencies because of the exploding and vanishing gradient issue [57]. The purpose of LSTM based recurrent neural network was to solve this issue and through the time LSTM proved to perform better than regular RNN. The Language model based on LSTM was proposed in this [58] work, where the structure of LSTM was almost the same as RNN except the neural network part. The very first LSTM based RNN was proposed by [57], but it was improved and gained a lot of popularity because of this [59, 60] works. The architecture of LSTM is the following:

$$\begin{aligned}
i_t &= \sigma(U_i \cdot x_t + W_i \cdot s_{t-1} + V_i \cdot c_{t-1} + b_i) \\
f_t &= \sigma(U_f \cdot x_t + W_f \cdot s_{t-1} + V_f \cdot c_{t-1} + b_f) \\
g_t &= \sigma(U \cdot x_t + W \cdot s_{t-1} + V \cdot c_{t-1} + b) \\
c_t &= f_t * c_{t-1} + i_t * g_t \\
o_t &= \sigma(U_o \cdot x_t + W_o \cdot s_{t-1} + V_o \cdot c_t + b_o) \\
s_t &= o_t * f(c_t) \\
y_t &= V \cdot s_t + M \cdot x_t + d
\end{aligned} \tag{3.31}$$

where, i_t, f_t, o_t are input gate, forget gate and output gate. The internal memory of the unit is $c_t \in R^{n_h}$. U_i, U_f, U_o, U , W_i, W_f, W_o, W and V_i, V_f, V_o, V are the weight matrixes. b_i, b_f, b_o, b and d are vectors of bias. $f(\cdot)$ is the layer's activation function. $\sigma(\cdot)$ is the activation function of the gates.

4 END-TO-END BASED ASR SYSTEMS

Because of the disadvantages of HMM based ASR systems mentioned above and popularity of deep learning area, a lot of studies and researches focused towards end-to-end large vocabulary continuous speech recognition (LVCSR). Figure 4.1 shows that the structure of an end-to-end model based on mapping the input audio sequence into word or phoneme sequence directly.

Almost all of the end-to-end models consist of following components: encoder, which is responsible for mapping the input sequence into feature sequence; aligner, which is responsible for providing the alignment between feature sequence and language; decoder, which decodes the final output.

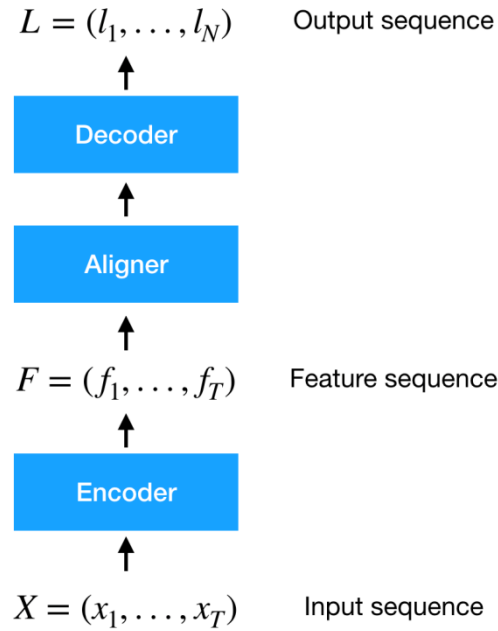


Figure 4.1 - End-to-end based model structure

However, this structure may not always be the same, since the end-to-end system is a complete architecture; it is usually hard to tell which component is responsible for what.

One of the main differences from traditional HMM based models which contains of several modules, the end-to-end model replaces it with deep neural networks that directly maps the input audio sequence into word sequence. Moreover, there is no need to make post-processing on the output.

Taking into account the above mentioned differences with HMM based models, end-to-end models for LVCSR get the following characteristics:

- Several modules can be merged into one network in order to joint training. The main advantage of merging modules is that there is no requirement of constructing many modules to map between intermediate states [61]. Joint training makes the end-to-end

model to use the function which is mostly connected to the final estimation as an optimization, allowing for search the global result [62].

- It makes the mapping between acoustic sequence and text sequence directly, without requiring the further processing step in order to achieve the right transcription [63]. In HMM based models, there are independent models that are representing the acoustical characters and pronunciation characters.

Looking at these advantages of end-to-end based models over HMM based models; it makes it simpler to construct the architecture for recognition speech.

However, sequence-to-sequence modeling tasks sometimes face data alignment issues and speech recognition task is not an exception. HMM based model cannot avoid the problem of aligning the label in the label sequence to the audio data as well as the end-to-end based models. Every frame of the audio relates to all possible states with probability distribution and doesn't have a requirement of forced accordance [62].

The end-to-end models according to their realization of soft alignment categorized into three parts:

- Connectionist Temporal Classifier (CTC) based: CTC initially estimates all possible alignments, and then it obtains the soft alignment by combination of these hard alignments. CTC makes an assumption that each label is independent from each other while estimating these alignments.
- Recurrent neural network (RNN) transducer: RNN-transducer also estimates all hard alignments and sums all of them in order to calculate the soft alignment. However, RNN-transducer compared to CTC does not make an assumption of label independence during hard alignment. It has a completely different approach in terms of paths and probability calculation compared to CTC.
- Attention based model: this technique does not need to estimate through all hard alignments. It directly calculates the soft alignment between input and output using method so called Attention mechanism.

4.1 CTC-based model

Nowadays the HMM-DNN based hybrid models have the state-of-the-art performance; the effect of DNN to overall system is limited. Most of the time, it is used to model the posterior probability of HMM and represents the local information. The feature within the time-domain is still handled by HMM. An attempt to replace the HMM to CNN or RNN to model the time-domain features fails every time, because it gets the data alignment issue. The loss function of RNN or CNN is interpreted at each place of the sequence which makes it to train, because it is crucial to know the alignment.

CTC was first suggested in [64]. Its development made it possible to take the full advantage of DNN in ASR and develop the end-to-end model. Basically, the CTC is simple loss function. However, it makes a big impact to overall training since it solves the

alignment issue while estimating the loss. CTC fundamentally overcomes two important obstacles:

- Alignment of data issue. There is no further need on manually segmenting and aligning the data. By solving this problem DNN can be used to model the time-domain features, which raises the importance of DNN to overall system.
- Outputting the transcription directly. It common that traditional models output the combination of phonemes and other elements, which requires an additional process to achieve the final target transcription. CTC neglects the need of these elements and outputs the direct target, which simplifies the overall construction very well.

By solving above issues, CTC with the help of a single network can do the mapping the input sequence to output sequence and execute the well-structured end-to-end system.

4.1.1 The core idea behind CTC

CTC method consists of two main processes: probability estimation of paths and aggregation of paths. Moreover, these two processes introduce a new blank symbol (“-”, meaning no label).

4.1.2 Path probability estimation

Having an input sequence $X = \{x_1, \dots, x_T\}$ with a T length, an encoder makes an encoding of input sequence into feature sequence $F = \{f_1, \dots, f_T\}$ with a same T length. f_t at time t is the feature vector whose dimension is one plus the size of the vocabulary ($f_t \in R^{|v|+1}$).

Using the softmax function, CTC turns the feature sequence $F = \{f_1, \dots, f_T\}$ into probability distribution $Y = \{y_1, \dots, y_T\}$, $y_t = \{y_t^1, \dots, y_t^{|v|+1}\}$, where y_t^a shows the probability of an output at the time t is a . $y_t^{|v|+1}$ shows the probability of blank symbol at t time.

If $V' = V \cup \{b\}$, V'^T describe the set of all sequences of length T that were described in vocabulary V' . Taking to account the description of y_t^k , we say that for a given input sequence X , the probability distribution of a particular sequence π is estimated as follows:

$$p(\pi|X) = \prod_{t=1}^T y_t^{\pi_t}, \forall \pi \in V'^T \quad (4.1)$$

where π_t defines the label at t of a π sequence. The component V'^T is known as a path and it is described by π .

After the calculation process is done, the input sequence will be mapped to a particular path with an identical length T . The conditional probability of a particular path will be estimated with an equation above. In simple words, this process means that for

every frame x_t there is a particular label π_t . This process of mapping an input to output can also be treated as hard-alignment procedure.

In the above mentioned equation (31) of estimation, we observe that it is based on significant assumption of independence. It means that the labels of the output are completely independent of each other. Any label selected at any time doesn't affect the overall distribution at other time. However, during the process of encoding, the y_t^k value and the future or past information of the speech are dependent on each other. In simple words, CTC makes an independence assumption on language model, not on acoustical model. For this reason, the CTC encoder is completely an acoustic model and it doesn't have an ability to model the language.

4.1.3 Path aggregation

During the estimation of probabilities of paths, we notice that the length of the output is the same as the length of input sequence, which is not always true. Usually, the transcription is much shorter than the speech sequence itself. For this reason, long-to-short mapping should be accomplished in order to aggregate these paths into short sequence.

We will denote the collection of label sequences with the length less than or equal to T as $V^{\leq T}$. The path aggregation will be denoted as a map function $B: L'^T \rightarrow L^{\leq T}$. This function makes the mapping between V'^T and $V^{\leq T}$. The aggregation process has two sub-processes:

- Merge the consecutive labels. If consecutive same labels stand by each other, we should combine them into one label. For instance, the path that has a label sequence “-d-oo-g-” or “-dd-o-g-” after the merging process gives the appearance “-d-o-g-”.
 - Removal of the blank symbol. The blank symbol should be deleted from the path, since it doesn't represent any label. The example above after this process gives the result “dog”.
- We notice that in above mentioned process, the paths “-d-oo-g-” and “-dd-o-g-” with the length 8 correspond to a label “dog” with the length 3. By analyzing the process, we can conclude that the particular label sequence may correspond to several paths. If we assume that the “dog” label has been obtained by aggregating the paths with the length 4, “dog” can correspond to different paths as shown in Figure 4.2.

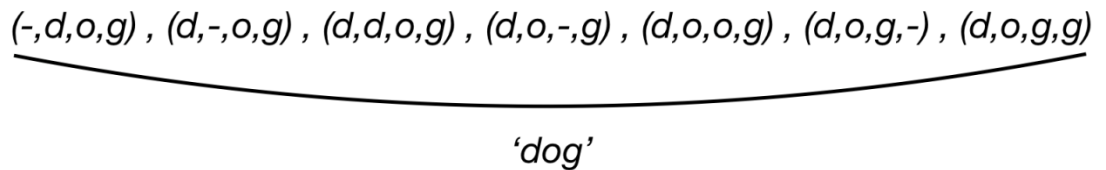


Figure 4.2 - Paths with length 4 for label 'dog'

The construction of paths for a corresponding label is shown in Figure 4.3. In the illustrated figure “-”, “d”, “o” and “g” are output values and 1,2,3,4 are the time steps. Following the arrows, path starting from the first time step and finishing with fourth time step, forms the label “dog”.

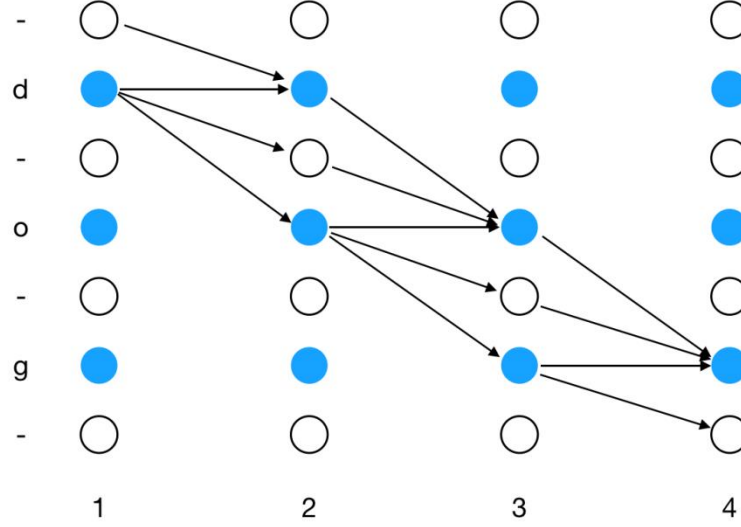


Figure 4.3 - Paths example for label “dog”

Other than achieving the label sequence from these paths, aggregation process also estimates the label sequence probability. Set of all paths corresponding to label sequence L is represented as $B^{-1}(L)$. Therefore, the label sequence probability given an input sequence X will be calculated as follows:

$$p(L|X) = \sum_{\pi \in B^{-1}(L)} p(\pi|X) \quad (4.2)$$

The estimation of $p(\pi|X)$ is shown in Equation 31.

It is clear that the calculation of $p(L|X)$ can be differentiated. Such a way, after achieving the label sequence probability, the model can be trained using back-propagation algorithm.

But, there is a hurdle on calculating the label sequence probability. It's hard to define how many paths out of V'^T are enclosed in $B^{-1}(L)$. Therefore, the calculation of the label sequence probability is mostly done by the forward-backward algorithm.

Despite the fact that the mapping process of input to paths is hard alignment, because of the path aggregation process, CTC doesn't make an input and output be aligned

by particular path. Therefore, we can say that CTC actually makes the soft alignment, which is completely different from HMM models.

The invention of CTC methodology drastically simplifies the structure and training process of large vocabulary speech recognition system. It doesn't require the building of extra dictionaries; it neglects the requirement of data alignment; it allows using any neural network structure with preferred number of layers to make the direct mapping between input and output.

4.1.4 Related works on CTC

One of the main advantages of using CTC is the absence of manual data segmentation. It makes the neural network like RNN or CNN play a significant role in the process. There has been number of models presented by the researches with various types of networks that have obtained very promising results in end-to-end ASR systems.

One of the first CTC related works on training the model for end-to-end LVCSR task was proposed here [65]. It was based on the network with three layers. First layer was the feed forward layer with 78 units; the second layer is LSTM layer with 120 units; the third layer is 27 neuron based LSTM. The training is handled by CTC layer. The conclusion of the work states that by increasing the depth of the network and the number of units in hidden layers, the performance can be improved significantly. It was confirmed by the results of [66] work.

Concluding the above researches, work in [62] stated that three layers based LSTM is too small for an experiment and it didn't give a desired result. This work based on CTC methodology trains the five layer based bidirectional LSTM (BiLSTM) with 500 neurons for each layer. The overall performance of this type of model nearly reached the state-of-the-art. Based on this paper many works related with 5 layer based networks appeared [67, 68, 69].

With a significant development of deep learning area, the CTC training based ASR models has also changed in term of the overall constructions and development.

In term of the network construction [70] proposed the CNN merged with RNN for speech recognition. The structure of the network consist of four layers CNN, two layers of DNN, two RNN layers and CTC layer. To avoid the problem of training difficulty when using RNN, [71] constructed a network that neglects RNN and instead uses CNN and CTC layers. The model designed with ten layers of CNN and 3 layers of full connection. Convolution operation was performed in frequency as well as in time domain. The outcomes showed more complex and deep networks were applied better the performance of the model has become. Moreover, they have concluded that well designed CNN layers can also model the time domain features.

In term of the complex network structure, [72] attempted to train the network with 9 layers, which contains seven layers of RNN. This model could exceed the human performance in some cases. In [73] they used CTC based network with 7 layers of

BiLSTM with 1000 neurons for each layer. The training was performed on 125000 hours of voice data from YouTube site. In the same way, [74] also used CTC based network with 9 layers of BiLSTM each having 1024 units that obtained promising results on their own dataset.

But, more we construct the deep and complex network to build an end-to-end ASR systems, doesn't always mean it will succeed in any occasion. A lot of recent researches [75, 76] used a simpler network (5 layers), because their dataset doesn't support training for large networks. For instance, [75] used only 5 layers of network for 2000 hours of data, whereas [73] used 7 layers for 125000 hours of data. The difference is very noticeable.

4.1.5 Training a large data

Many researches indicate that deep and complex networks require a lot of data and speech recognition area is not an exception. Well working ASR systems take a huge benefit from large-scale data.

To properly train the network with 5 layers, [67] used the datasets that have 7000 hours of clean speech, 5000 hours of their own data along with other data sources like Fisher, Switchboard etc. Combined with other noisy speech data they have obtained in total 100000 hours of speech data for training. Using these large datasets, researchers have achieved a very great performance on their noisy speech, which as a result outperformed a lot of huge companies like Google, Apple and so on. In their following papers, [72] they used 12000 hours of English speech data and 9000 hours of Chinese speech for training the 9 layer network. They have obtained better result than human.

In the base of Google, [73] they trained the 7 layer BiLSTM on 125000 hours of data and 100000 data of transcriptions. In their different experiment [77], they used 125000 hours of google voice search traffic data as well.

Very huge amount of data has a significant effect on performance of the model. However, using so much data requires also a significant improvement on training technique.

To make the training process faster on large datasets, [67] made an approach with two details: data parallelism and model parallelism. In data parallelism, the system used graphical processing unit (GPU) parallelism mechanism, where samples being processed using GPU in parallel. The parallelism was also realized between GPUs, where each GPU processes their own mini-batches and later merges their result together. Model parallelism, improves the RNNs calculation speed by dividing the tasks into time series and run, respectively.

In the works [67, 77] they constructed the effective methods for training large datasets more fundamentally: open message passing interface code was created to merge the gradients from different GPUs, an effective CTC calculation was proposed to run on

GPU, new memory allocation methodology was created. Overall, the model obtained 4-20 times faster training.

4.2 RNN-transducer based end-to-end model

There are several drawbacks of using CTC that affects its effectiveness negatively:

- CTC based systems can't model the dependency between output sequences, because it's based on an assumption that all units of an output are independent of each other. For this reason CTC lack the ability to lean language model. Therefore, speech recognition system trained using CTC is an acoustic model.
- CTC makes the mapping of input sequences to output which are shorter than the input. For cases, where output is longer than input CTC is completely not effective.

To model the dependencies between output elements is very important and makes a huge impact on speech recognition system. The paper [78] presented a RNN transducer based model. It constructs a new mechanism that focuses on solving the above problems of CTC. It can map an input sequence to output sequence with any length. Moreover, the dependencies between output variables can be modeled as well.

4.2.1 Main ideas behind RNN-transducer

RNN-transducer based model in term of the structure share a lot of similarities with CTC based model. For example, they both share the same loss function; they both solve the problem of manual segmentation between input sequence and output sequence; they use a “black symbol” element; they both estimate the probabilities of all paths and aggregates paths to obtain the label sequence. But, path generation and path probability estimation processes are very different. Here the advantages of RNN-transducer come to play.

RNN-transducer based model contains three important components: Network for transcription ($F(x)$); prediction network ($P(y, g)$); joint network ($J(f, g)$). The construction of RNN-transducer model is illustrated in Figure 4.4.

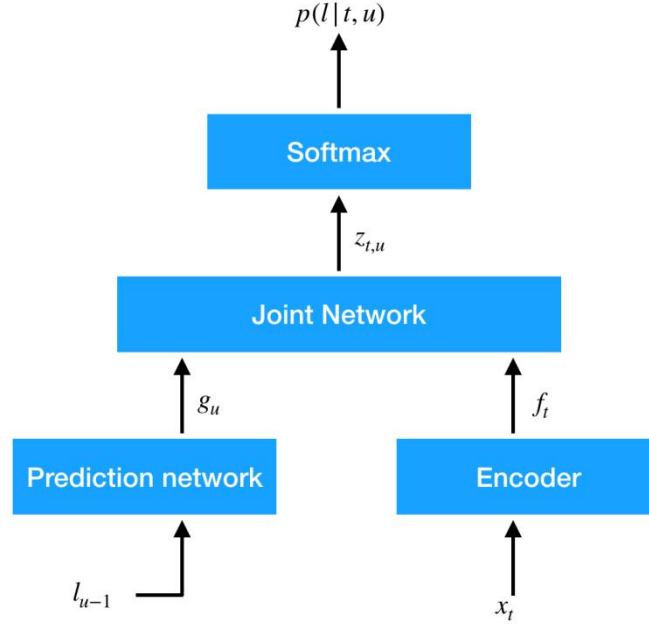


Figure 4.4 - RNN-transducer structure

These three components are based on their own function:

- Transcription network ($F(x)$). The role of this component is an encoder for an acoustic model. Given an input sequence $X = \{x_1, \dots, x_T\}$ function $F(x)$ maps it to features $F = \{f_1, \dots, f_T\}$. Accordingly, for value x_t , transcription function is played as an acoustic model resulting the output with $|v| + 1$ dimension size.
- Prediction network: This component works as a decoder, which models the language. While a transcription network models the dependencies inside acoustic input, prediction network models the dependencies within output sequences. In any location of the label $P(l)$ support the hidden state h_u and an output g_u . Calculation process of these components is the following:

$$h_u = H(W_{ih}l_{u-1} + W_{hh}g_{u-1} + b_h) \quad (4.3)$$

$$g_u = W_{ho}h_u + b_o \quad (4.4)$$

Analyzing the joint network, we will find out that the estimation of l_{u-1} directly depends on g_{u-1} . In simple words, the above equations mean that using $l_{[1:u-1]}$ sequence defines the l_u .

- Joint network ($J(f, g)$): it handles the alignment problem between input sequence and output. For $t \in [1, T], u \in [1, N]$, the network takes the output of transcription network and the output of prediction network in order to calculate the label distribution:

$$e(k, t, u) = \exp(f_t^k + g_u^k) \quad (4.5)$$

$$p(k \in V' | t, u) = \frac{e(k, t, u)}{\sum_{k' \in V'} e(k', t, u)} \quad (4.6)$$

It is clear that $p(k \in V' | t, u)$ is f_t and g_u based function. Knowing that f_t is obtained by x_t and g_u is obtained by the sequence $\{l_1, \dots, l_{u-1}\}$, the joint network does the calculation of the label distribution $P(l_u | \{l_1, \dots, l_{u-1}\}, x_t)$ at location u .

In the base of $p(k \in V' | t, u)$, there are individual way of constructing the path generation and path aggregation by RNN-transducer and it has an according algorithm to estimate the probability of a particular sequence [78]. Since there is an existence of the process of path aggregation, RNN-transducer is designed as a soft alignment technique that doesn't require any explicit data alignment.

The process of decoding by RNN-transducer is the following: for every input x_t into the model, it will output the label until it faces the blank symbol “-”. Once this symbol is faced, it continues to read the next element x_{t+1} . This whole process is repeated until all input are read. Since the input x_t has the ability to generate several labels, RNN-transducer can easily overcome the cases in which an output sequence length is greater than input sequences.

Taking into account the above ideas that lay behind RNN-transducer we can make the following statements:

- Because of the ability of the input to result in any length label sequence, it means that RNN-transducer can map the input sequence into output sequence with any length.
- The prediction network based on RNN structure. This means that each state based on previous state and output. This way RNN-transducer can learn the dependencies inside the output sequence, which means it do a language modeling.
- Joint network applies acoustic model as well as a language model to calculation the label distribution probability. For this reason RNN-transducer can model the dependencies between input and output sequence by joint training the acoustic and language models.

This paper has been trained and tested on TIMIT corpus of speech data [79] and achieved 23% PER (phone error rate), which considered as the best outcome for RNN based systems.

4.2.2 RNN-transducer related works

In order to improve the performance of RNN-transducer, the work in [44] has made adjustments for further enhancements. They switched the joint network from a simple addition to a full connection layer. They have increased the complexity of the network and pre-trained the prediction network and transcription network. As a result, they have

achieved a 17% PER on TIMIT data, which was huge leap forward. The comparisons in the experiment showed that it is hard to train the RNN-transducer from zero. Pre-trained RNN-transducer could be more effective in model performance.

The paper in [80] has explored the problem much deeper and enhanced the RNN-transducer in different ways. They constructed the model for transcription network with 12 LSTM layers and a prediction network with 2 LSTM layers. However, it does pre-training with CTC in order to output words, fonts and phonemes at 12th, 10th and 5th layers. Moreover, it doesn't output the word as a unit, but outputs the sub-word unit. As a result, they have achieved 8% WER on Google Voice Search data.

Event that RNN-transducer has considerable benefits over CTC, these benefits can also lead to some limitations. One of these issues is the appearances of the unwanted paths. Since RNN-transducer allows an input sequence to generate multiple output sequence, it is obvious that this issue may appear. For instance, one frame of an audio may generate all outputs, while other frames produce empty labels. These unreasonable paths in the process are definitely an exception in RNN-transducer case. The address this issue, papers [81, 82] proposed the recurrent neural aligner. The idea of this proposal is the limitation to RNN-transducer where for an input there is only one output. However, it didn't solve the overall issue.

Exploring all the related work on RNN-transducer based models it appears that even it has noticeable improvements over CTC based models, RNN-transducer has a following difficulties:

- Analyzing a lot of experiments it clear that RNN-transducer is hard to train. For this reason, it is crucial to pre-train all of the components separately, which ultimately can lead to considerable improvements in terms of performance.
- RNN-transducer makes a lot of unnecessary calculation on many unwanted paths. Although, there are number of works on addressing this problem, the conclusions state that it is not possible to avoid this issue. However, all ASR related tasks that calculate the possible paths and aggregate these paths meet this problem.

4.3 Attention based model

The attention based model first was proposed in [83] on the machine translation problem. Their aim was to solve problem of encoder-decoder model on machine translation requiring encoding the text into fixed sized vector, which makes its encoding ability be miserable. In the paper, the encoder encodes the input into sequence of vectors, whereas the decoder with the help of an attention mechanism at every output assigns different weights to each vector in the sequence. By the weighed summation and historical output, the next output is defined. This technique overcomes the issue of encoding all information into fixed sized vector, in order for long sentences to have a well effect of encoding. This paper has made a great effect in the field of speech recognition task in the following manner:

- Speech recognition is also considered as a sequence-to-sequence task, where an input maps to an output. It is quite similar to machine translation problems
- The technique of attention encoder-decoder doesn't require a segmentation of the data. Using attention, it can easily model the soft alignment between input and output sequences, which leads to a big step forward in speech recognition
- The result of encoding process is not limited on fixed sized vectors, which means that it can have a good impact on sequences with long lengths. It makes it possible to process speech recognition with different lengths.

Looking at the above advantages, attention based end-to-end models gained a lot popularity and interest.

Attention based model can have three different components: encoder, aligner and decoder. The attention mechanism is used by aligner component. The construction of the attention based model is illustrated in Figure 4.5. Each component can be enhanced separately.

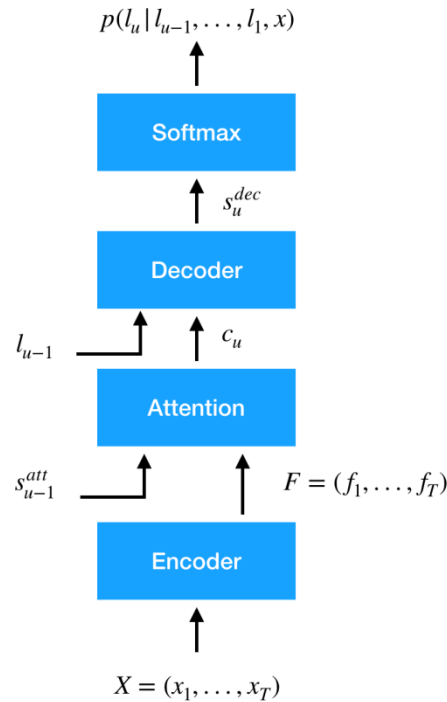


Figure 4.5 - Attention based model structure

The acoustic model is realized by an encoder, in the same way as CTC and RNN-transducer based models. The problems that attention based model can face is the same as in CTC and RNN-transducer. However, the attention-based encoder may face some different issues.

4.3.1 Redundancy of the information and delay

A valuable issue that can be faced by the mixture of an encoder and attention mechanism is latency. Attention is used to an entire outcome of an encoding, which means that before starting, it needs to wait until an encoder finishes the whole process. In contrast, this kind of issue is not the case for CTC and RNN-transducer based models. More time encoding process lasts, more delay happens in model. Moreover, if the encoder doesn't decrease the length of the sequence, it will produce the result with much longer length than the targets (input speech sequences usually longer than transcriptions). For this reason two issues may occur: first, too much calculation for attention since the result sequence can be long, therefore more delay time; second, the resulted sequence produced by an encoder without any sampling method can generate a lot of unnecessary (redundant) information to an attention. Machine translation problem presented in [91], doesn't take into consideration the length of the sequence issue, which can happen in speech recognition. Moreover, works in [83, 84] that were based on attention mechanism for automatic speech recognition system didn't mention this problem.

The work in [85] was first to express and address the problem. They have come with this work based on the previous one [84]. They used the pooling method on several RNN layers to make the model faster. Pooling method is applied in time dimension, which results the less length of input sequence. The time step of an encoder is reduced by $\frac{1}{2}$ between first and lower layers. This reduction is handled using sub sampling or other methods. The encoders structure is 4 layer of bidirectional gated recurrent unit (GRU) [86], in which the last 2 layers read the result of a previous layer each time step. This lead to a significant $\frac{1}{4}$ input sequence reduction compared to an original one. The technique is called sub sampling.

For the same reason of reduction of an encoders sequence's length, listen, attend and spell (LAS) in [87] proposed another methodology. The encoding component is called listener consisting 4 BiLSTM layers. To reduce the steps of RNN, each layer takes as an input the previous layers consecutive 2 time steps. Such a way, the steps of RNN can be reduces to $\frac{1}{8}$ proportion. This structure of RNN is called pyramid structure. After this, most of the experiments and works have applied to an encoder the pyramid structure. Some of the works have reduced the length of an input to $\frac{1}{4}$, whereas several works reduced the length down to $\frac{1}{8}$ proportion.

Moreover, there is a methodology that reduces the input sequence even before encoder is operated (applied directly to an input) [88, 89]. Using these reducing (sum sampling) techniques, the length of the encoder's outcome is decreased, which somehow avoid the delay issue and redundancy of an information issue in attention mechanism.

4.3.2 The structure of a network

In order to enhance the encoding capability, the attention based encoder-decoder model is becoming more complex, similar to CTC and RNN-transducer based models development. The most obvious complexity reflects on the depth of a network. An initial structure of an encoder contained 3 layers, then slowly developed to 4, 5, 6 layers [90, 89]. As the network becomes more complicated in terms of depth, the performance of the model improves as well. The work in [91] structured the network of an encoder with 15 layers with batch normalization, convolutional LSTM, residual network. As a result, they have obtained on Wall street journal dataset 10.5% WER (Word Error Rate) with no external language model.

4.3.3 Related works on Attention based model

In terms of functionality, the attention mechanism is divided into 3 components: location based, context based and hybrid. The process of estimation of their attention weights are described in Equation 36. They are different in terms of usage attention processes information and in terms of characteristics.

- Context based uses the previous hidden state and an input feature in order to calculate the weight for every position. The problem of this method that it doesn't use position information. It will give the identical value for features with different characteristics in feature sequence. This issue is called similarity fragment problem.
- In the location based part the previous weight is used as information of location in order to estimate the current weight. It is not useful for input features, since it doesn't use it.
- Hybrid part takes an advantage of context and location based attentions. It takes into consideration the previous weight, previous hidden state and input features to make the calculation:

$$\alpha_u = \begin{cases} \text{Attend}(s_{u-1}, f) \\ \text{Attend}(s_{u-1}, \alpha_{u-1}) \\ \text{Attend}(s_{u-1}, \alpha_{u-1}, f) \end{cases} \quad (4.7)$$

Based on these different approaches in terms of structure and functionality of an attention method, there are different variety of problems in speech recognition that can be solved.

4.3.4 The problem of continuity

When it comes to a machine translation task, it is a normal circumstance that 2 neighboring words in the result sequence may be placed in the completely different locations of initial sequence and correspondingly 2 completely far from each other words may be appearing in original sequence next to one another. However, this situation doesn't

appear in speech recognition task. The production of the sound states that 2 neighboring words in the transcription should be corresponded to a 2 neighboring parts in the audio. This phenomenon is called continuity of the speech recognition. To solve the problem of continuity location based or hybrid attention method must be constructed.

Based on [83, 92], some enhancements have been made to solve the continuity issue. Following expressions are enhanced process of attention mechanism:

$$e_{u,t} = a(s_{u-1}, f_t) \quad (4.8)$$

$$\hat{e}_{u,t} = d(t - E_{\alpha_{u-1}}[t]) \exp(e_{u,t}) = d\left(t - \sum_{t=1}^T \alpha_{u-1,t} t\right) \exp e_{u,t} \quad (4.9)$$

$$\alpha_{u,t} = \frac{\hat{e}_{u,t}}{\sum_{t=1}^T \hat{e}_{u,t}} \quad (4.10)$$

$$c_u = \sum_{t=1}^T \alpha_{u,t} f_t \quad (4.11)$$

We can see here that $a()$ and $d()$ are the functions that should to be learned during the training. It is clear that, Equation 37, 38 utilize s_{u-1}, f and α_{u-1} in their expressions, which leads to hybrid attention mechanism. In comparison to context based, the expression 38 uses $d(t - E_{\alpha_{u-1}}[t])$, which provides a penalization of an attention weights depending of an offset between previous and current position. In this, they solve the problem of continuity.

4.3.5 Monotonic problem

The monotonic problem is often associated with continuity issue and it states that sound is one directional in time domain. Once we found out the part of an audio that corresponds to some particular label, it is nearly impossible to match this part to another label. In simpler words, if the audio part has been corresponded to particular label through attention, this part of an audio should be taken out of the consideration.

Particularly, one label may show up multiple times in different places of a transcription. These labels should be corresponded to different audio pieces. But, in calculation time it is possible that they may attend on the same audio piece. This issue can be addressed using attention's location information.

[92] proposed a solution that consists of a new loss terms in training:

$$p_u = \max\{0, \sum_{t=1}^T CDF(u, t) - CDF(u - 1, t)\} \quad (4.12)$$

where, $CDF(u, t) = \sum_{j=1}^t \alpha_{u,j}$ is a cumulative distribution function of the label at u position and t time step. $\alpha_{u,j}$ defines the feature weight at time j to produce the position of a label u . It is obvious that, earlier attention weight is, more times it gets estimated in p_u .

4.3.6 Key information extraction problem

Earlier [83] noticed that the attention based systems are only targeted to recognize the speech data with an approximately same length. Once the length of an audio is noticeable longer than training data, the models performance significantly drops. In [84] authors claim that the reasons behind this issue are the following: first, lack of location information usage in attention mechanism; second, the lack of ability of an attention mechanist to extract valuable key information from features.

The first issue, [84] constructed the following attention with the help of location information:

$$s_i = Recurrency(s_{i-1}, c_{i-1}, l_{i-1}) \quad (4.13)$$

$$h_i = F * \alpha_{i-1} \quad (4.14)$$

$$e_{i,j} = \omega^T \tanh(Ws_i + Vf_i + Uh_{i,j} + b) \quad (4.15)$$

$$\alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{j=1}^t \exp(e_{i,j})} \quad (4.16)$$

$$c_i = \sum_{j=1}^T \alpha_{i,j} f_i \quad (4.17)$$

In the above equations 4.13-4.17, by inference we can see that $\alpha_i = Attend(s_{i-1}, \alpha_{i-1}, f)$ is the hybrid mechanism. However, to make an operation on location information, compared to simple hybrid attention, it utilizes convolution (Equation 43).

To address the second problem [84] stated that the reason behind the failure of an attention mechanism to focus on valuable and useful information of the feature is the following properties of a softmax function, which is used to calculate the weights:

- Every element of the weights, which were estimated by softmax is greater than zero, which that every feature is remained in the context achieved by an attention. Therefore, this provides an unwanted noisy information
- Softmax focuses the probability on large elements, which causes the attention to concentrate on one feature, which leave other features that have significant effect not fully retained.

Moreover, because of the weights of an attention should be calculated for each feature in the sequence at every step, it lead to a big amount of calculation. To solve this issue, the following methods are proposed:

- Expanding the greater weights and shrinking the small ones, by providing a scale value in softmax which is greater than unity. This technique fades the noise characteristics issue. However, it makes the probability distribution focus on the elements of the feature with probability that is higher.
- It saves the biggest value and normalizes the weight of a softmax after calculating them. This technique raises the calculation amount of an attention
- The weights of an attention are estimated only for data in window, which is calculated and with window sliding method set beforehand. This technique is called a window method.

The third method is used in the process of inference, not training. After the experiment, the result showed 17% WER on TIMIT dataset. For the 10 times expanded in length sentences, it still showed pretty promising result of 20% WER.

The authors in [85] made a further improvements based on above techniques. However, this time window method was utilized not only in inference process, but also in training process. In addition, the window method differs from the one that was in [84]. In [92], the selection of a window is dependent on the previous window's quality. This makes the model train hard at initial states, because it can lead to a local minimum. The correction in [85] attenuates this dependency. The window initialization in [85] is handled according to a speaking speed of data and blank voice length, which lead to a better and effective performance.

4.3.7 Delay

There 2 reasons that cause delay. First one happens when result sequence is long, which can be reason for decoder to be slow in computation phase. The effectiveness of this issue has been weakened using sub-sampling. Second one is related to attention's action on entire result of an encoding. This means that attention cannot start to take an action until the complete result of an encoding process is produced. There were a lot of attempts to address this problem by researchers.

To solve this issue related to delay, authors in [93] have built an online speech recognition system. Although, it was based on one directional RNN encoder, this system utilized the window mechanism in [85]. The benefit of using this technique (windowing)

is that attention doesn't have to wait until the process of encoding is finished. Once it has enough information to cover the window, it can begin the process. Therefore, the delay problem of an attention can be taken into control.

The window gets calculated first at some position u using the following equations:

$$m_u = \text{median}(\alpha_{u-1}) \quad (4.18)$$

$$W_u = \{f_{m_u-p}, \dots, f_{m_u+q}\} \quad (4.19)$$

In attention operation the model takes into consideration only f_i that shows only inside the window W_u :

$$s_u = \text{RNNDecoder}(l_{u-1}, s_{u-1}, c_{u-1}) \quad (4.20)$$

$$e_{u,j} = v^T \cdot \tanh(\psi(s_u) + \varphi(f_i) + b) \quad (4.21)$$

$$\alpha_{u,j} = \frac{\exp(e_{u,j})}{\sum_{j=m_u-p}^{m_u+q} \exp(e_{u,j})} \quad (4.22)$$

$$c_u = \sum_{j=m_u-p}^{m_u+q} \alpha_{u,j} f_j \quad (4.23)$$

With the help of window technique, [94] believed that using a fixed window size is not reasonable for a real situation. Therefore, they have proposed a new window technique based Gaussian prediction. The slide step and the size of a window are estimated based on the state of the system instead being preset beforehand. This approach can be adaptive to any kind of change in terms of speed.

The calculation of window using Gaussian is as follows:

$$s'_u = \text{Recurrent}(s_{u-1}, l_{u-1}) \quad (4.24)$$

$$\Delta p_u = S_w \cdot \text{sigmoid}(V_p^T \tanh(W_p s'_u)) \quad (4.25)$$

$$p_u = \Delta p_u + p_{u-1} \quad (4.26)$$

$$\sigma_u = D_w \cdot \text{sigmoid}(V_\sigma^T \tanh(W_\sigma s'_u)) \quad (4.27)$$

$$e_{u,j} = \begin{cases} \exp\left(-\frac{(j-p_u)^2}{2\sigma_u^2}\right) & j \in [1, \text{floor}(p_u + K\sigma_u)] \\ 0, & \text{others} \end{cases} \quad (4.28)$$

$$\alpha_{u,j} = \frac{e_{u,j}}{\sum_{j'=1}^T e_{u,j'}} \quad (4.29)$$

$$c_u = \sum_{j=1}^T \alpha_{u,j} f_j \quad (4.30)$$

where, p_u is the mean and σ_u is the variance of Gaussian distribution. They are reflecting the position and the size at u position. It may look like the estimation of $\alpha_{u,j}$ happens on the whole result of an encoding. However, $e_{u,j}$ outside the window is equal to 0, which significantly decreases the delay and executes a dynamic window.

4.3.8 Decoder

The decoder component usually based on RNN gets the context information produced by an attention, then decodes into transcription. In term of structure, the decoder can work in 2 different modes. First one using a previous state executes the attention mechanism on the result of an encoder in order to obtain the context information, and then calculates the decoder's current hidden state. Second one first estimates the current state of decoder, after that computes the context information using attention mechanism on result sequence. The second one has more popularity than the first one [93, 91, 85].

Number of works attempted on enhancing the attention-based speech recognition system performance in terms of decoder.

In [88] authors discovered that the decoder contains only one-layer and weren't structured for features of language. Therefore, they have structured the decoder as follows:

$$p_i = f(l_{i-1}, p_{i-1}) \quad (4.31)$$

$$s_i = f(p_i, s_{i-1}, c_i) \quad (4.32)$$

$$p(l_i | l_{i-1}, \dots, l_1, c_i) = g(s_i, c_i) \quad (4.33)$$

In comparison to simple decoder structure, in [88] they have added the layer based on an equation above. It works on historical output, which means that it completely acts like a language model. This allows the decoder to work on longer terms.

Authors in [95] have constructed a multi-head decoder technique. In this technology, for extracting the context information of encoding result and decoding, different attentions are used. The outcomes of these attentions are merged into one final output.

The attention technology has been greatly learned and studied in term of human perception of the world. For any occurring problem, a new solutions and analysis are appearing. But, many experiments done by variety of researchers show that attention based models can produce a lot better performance compared to CTC and RNN-transducer based models.

4.4 Summary

The comparison and conclusion of 3 well known end-to-end based models in terms of several features is the following:

- In terms of delay, the training algorithm in CTC can estimate the probability of decoder output without waiting obtaining the encoding result, which means that CTC has a very short delay time. RNN-transducer on the other hand can be considered to be similar to CTC. However, the delay time in RNN-transducer can be higher compared to CTC, because it utilizes not only encoder result, but also the decoder output of the previous step in forward-backward algorithm. They both support streaming recognition, which means that the result is produced simultaneously with speech input. The attention based model happens to have a long delay time, since it has to wait until the entire result of an encoder is generated before it can make an alignment and decoding. The streaming recognition is not the case here.
- Complexity in computation: while estimating the probability of length N based on features with length T , CTC has the $O(T)$ complexity, since it has to perform a softmax operation to each time step. RNN-transducer requires to perform a softmax on each input and output part, which gives the complexity of $O(TN)$. The traditional attention mechanism, similar to RNN-transducer, requires to perform an operation on input and output part, that gives the complexity of $O(NT)$. But, as an attention technology utilizes a window technique, its complexity gets reduced, since it needs to perform an operation on inputs within a window.
- Ability of language modeling: Since CTC makes an assumption of input and output being independent it doesn't need a language model. Models based on attention mechanism obtain this ability using a decoder, while RNN-transducer achieves the language modeling capability using joint network.
- Capability of training: Basically, CTC is considered as simple loss function, which means there are no weights to train. The only role of CTC is to train an acoustic model. Because of this, CTC can easily reach to optimal outcomes. The models based on RNN-transducer and attention technology have many functions and weight that need to be trained. Since there is a joint training of acoustic and language models, it is much more complex to train

them compared to CTC. Moreover, the joint network based on RNN-transducer produces a lot of unwanted alignment of data between input and output. Therefore, it even more complex to train an attention based systems. Usually, it needs to have pre-trained transcription and prediction networks in order to achieve a promising performance.

- An accuracy of recognition: The accuracy of RNN-transducer and attention mechanism is a lot better than simple CTC. However, the performance of attention based model usually is better than RNN-transducer based model.

5 SPEECH CORPORA IN KAZAKH LANGUAGE

Speech data in most of the languages that have a low resource usually doesn't even exist. Therefore, producing speech corpora is very challenging and requires tremendous amount of time. Kazakh language due to its lack of popularity considered to be low-resource language. This section provides an overview on many data collection techniques, marking some of the issues related to a particular method. The main aim of this section is to present crowd sourcing web-based tool called "Kazakh recorder" which accessible on the website and designed to make the collection of Kazakh speech data more conveniently and quickly. Moreover, this section provides a statistics of people (age, gender, number of sentences) who made a contribution on collecting this speech data. Using this tool, we have collected over 100 hours of speech data 65 different native speakers, each having on average 500 sentences pronounced in Kazakh language.

5.1 Introduction

The communication between humans can be in different forms like speech, pictorial language, gestures, sign language etc. Among these forms, communication using speech is considered to be more effective and popular. This leads to the fact that human computer interaction should be handled using speech communication. Therefore, this fact highlights the importance of developing Automatic Speech Recognition system.

The performance of ASR system is directly dependent on the quality of speech data. However, speech corpora can be based on general language or domain specific language. Using a dataset which is domain specific has the benefit of growing the capacity of recognition process. Moreover, utilizing the real speech of users as a dataset raises the ASR's overall productivity.

It is very significant and crucial factor to find the adequate speech dataset while constructing the ASR system for low-resource languages like Kazakh. However, the non-existence if required amount of speech data for low-resource languages is obvious. Thus, the tools and instruments for collecting the speech data can play a significant role in building speech recognition systems.'

Collecting a speech data in order to construct a speech corpus can be executed by searching from the web sources of already existing datasets [96, 97] or by gathering a completely new speech records from scratch using different methodology. The first one is for collecting a generic speech data, while the second one is for collecting a domain specific data, which can be handled using various methods. The collection of a new speech data can be executed in local conditions, requiring speakers be in the environment [79, 98]. The benefit of this technique is that it gives the ability to control the conditions of an environment, excluding the factors like noise, disturbance and external impacts on current speaker. However, this technique is not effective when you have large number of speakers, because it is dependent on physical presence of the speaker in given environment, which

can cause various difficulties. In contrast, collecting the data remotely using phone calls [99, 100] gives the ability to speakers to contribute on collecting the data independent on their current geographical position. Compared to the previous technique, this method can a lot more effective. However, disadvantages of this technique can be unsupervised contribution, not controlled conditions and constant noise presence in the telephone devices.

Since the word wide internet is considered to be current trend, speech data collection techniques and methods are usually based on websites and mobile applications [101, 102, 103]. This technique is considered to be as the most efficient for a big amount of users despite its lack of ability to control the current environment.

Kazakh is the Turkic family language and it is considered to be low-resource language. About 22 million people are speaking in native Kazakh language. The early attempts on building a speech recognition system in Kazakh language have been made here [104].

We are proposing a web-based data collecting instrument which focused on speech data collection of low-resourced languages. The simulation of abilities of this web tool is handled by collecting the speech data of Kazakh language. The explorations and analysis of an outcome indicate that this technique gives the ability to collect the desired amount of speech data painless and in short amount of time. The outcomes learned in this method of data collection can be useful for other researchers who are planning to utilize the same approach.

We have chosen several different books from Kazakh literature for a literature domain implementation. These books are written by a famous Kazakh writers like Abay Kunanbayev, Beyimbet Maylin, Berdibek Sokpakbayev etc.

5.2 Related work

Collection of data is the most significant and important part in building any recognition task. Exploring many literatures and papers, we have discovered several options on collecting speech related data:

- The way of recording directly [105, 106]
- Conversations through telephone [107]
- Resourcing via web [96, 97]
- Questionnaires responses through telephone [108,109, 110]

Direct recordings are executed in local preset conditions by the help of people who are aware and involved in collecting activity directly. Usually, these conditions are provided by professional studios that are capable of controlling and excluding unwanted noises to produce a clear speech voice. But, these kinds of technologies usually suffer from a lack of speaker and completed with a very few speakers. For this reason, this technique takes a large amount of time to achieve a large corpus of speech data. This work

[107] claims that ASR systems that achieved well performance on local environment works less productive when it faces real world data.

Collecting a speech data using telephone conversations can be considered as more effective way of data collection compared to direct recording. In this work [107] special recording equipment was attached to a simple telephone operator workplace and the data was gathered. But, this experiment was conducted a very long time ago. If an identical approach was conducted in modern days, the data collection would be handled using call centers. As we know, it is nearly impossible to use these recording that have been collected in such an environment, unless it is targeted for this call related workplaces. As long as mobile operators not willing to share their collected speech recordings or not giving permission to use them, building a speech corpus using this technology is not possible.

Questionnaires responses through telephone are considered to be an expensive method of data collection. Because of the low funding for the most researchers, this method is not the most affordable way to obtain a complete speech corpus.

Although the telephone based data collection method is more preferable than data collection in studios, it not considered as the most practical data collection method. There are several drawbacks in this technique like uncontrollable environment for the speaker, ability of the user to speak, external noise and user identification. These drawbacks are listed in [111].

There is a lot of literature related to building a speech corpus using a web-based resourcing. The sources for a web-resourcing method can be considered Radio, TV, and Social Medias like YouTube etc. Although, there are millions of audio recording that can be parsed from these sites, this technique is not suitable for a specific domain. If the dataset is loaded from these sources to a specific area, to find the exact recording to this area is almost impossible. In addition, in order to train the ASR system properly, it needs a big number of utterances, which is required to be pronounced by different speakers. Assuming that all the recording from these sources are downloaded, we can say that there are still a lot of works to be done on building a transcriptions of these audios and text corpuses. Obviously, this work takes long hours. Since the invention of Internet, a lot of datasets can be crowd-sourced, but there are many difficult obstacles to overcome [112].

There is another method related to web based approach is to gather the speech recording using web application [113]. Usage of this method provides an ability to predefine the transcriptions and present them to speakers via web application in order to obtain their corresponding recordings. Moreover, this technique doesn't require physical presence of the user in local conditions and makes possible to record a different types of speech data with different speakers as well. Nevertheless, dependent on a computer which produces variety of recording and uncontrollable noisy conditions can be considered as potential obstacles. However, the benefits of this technology shouldn't be underestimated, especially when we live in the world with an easy internet access.

Exploring literatures, we have discovered that there are systems which enable data collection. However, these systems cost is very high and for not popular languages like Kazakh is not suitable.

The very recent and trended methods are to utilize mobile devices to gather the data in order to construct the corpus. Because of the popularity and convenience of mobile devices all over the world, it is more suitable and effective way to handle the speech data collection. The work in [101] has made an application on iOS platform to collect the speech recordings. The transcriptions of sentences are presented in application so the speaker can record them via microphone. Very interesting ability of this application is that it can process offline as well. Offline recordings are saved in local memory of the phone unless it gets the access to the internet. The similar approach was provided later with application which is based on Android platform [114].

Although these techniques have many benefits over other above discussed systems, some issues and obstacles cannot be avoided. The main problem is uncontrollable environment for the user. It is impossible to influence the quality of an environment for the user, which leads to some possible recordings with external sounds and noises. Therefore, because of the inability to control and monitor the environment and quality of the recording, we are manually listening each recording and based on the quality of it we make the decision whether to delete the recording or remain it. This is only handled by admin user who is in charge of adding books and controlling everything. Using this ability we can solve some issues which are present in previously discussed methods of data collection. The speech data collection operation gets a lot quicker and effective, because almost every average person has a smartphone with a proper internet connection. Another important ability of this technique is independency of specific domain, which is provided by adding any text transcription to collect the speeches.

The researches in the past have already considered using mobile applications, but there are been some restrictions related to the phones type. Because of this, our web-based application has a mobile version of itself, which make it entirely independent on the version or type of a smartphone.

5.3 Methodology

During the process of developing this instrument, our methodology was based on predefined design regulations. Based on these regulations we have designed set of rules, and then we have designed the architecture itself which ultimately becomes capable of satisfying these rules. We also made a selection on technology stack that supports these rules and regulations and makes the development process easier. For demonstration purposes, we will be describing the every single process of using this tool.

5.3.1 Regulations

Since the design regulations define the aims and expectations from the overall data collection process, it is considered to be very important step. The architecture of our instrument was constructed based on these regulations that are listed as follows:

- Portability: it is considered to be the most important and significant case of collecting the data. It is better than getting together a lot of people in one place and gathering the data using local equipment. Website development or mobile application development is more productive and effective in terms of portability. Despite the fact that professionally equipped studio generates far better quality recordings for speech recognition systems, it is considered to be impractical and not preferred in most cases.
- Post-processing simplification: In the process of developing a large corpus, the datasets should go through post processing stage. For example, matching the names of transcriptions and audio files, structure the folder system in the way where speakers are separated and saving the audio recording in particular format that is targeted in neural network processing
- Maximum recording capacity: With the help of a mobile phone and website for data collection makes it possible to obtain maximum amount of records. Creating a website which is adapted into any smartphone itself and its screen size, makes this method available for any target audience to gain a maximum possible amount of recordings.
- Usability: Usability of an application is a major part of overall success of data collection. This data collection technique based on web application will be utilized by people of different gender, age, social status etc. For this reason, it is very significant to have a tool with intuitive functions and ease in usability. For it cover a lot of people to gather the data and not causing questions by users, it is important to insert instructions within the instrument.
- Multilingualism: An application supports multiple languages. The backend of the website can enable different languages. Therefore, it can be used as a tool to collect speech data for any language.
- Open-source: One of the important aims of technology is to construct an instrument which is fully customizable. Such a way, any researcher can customize this project to his own needs in order to collect data for low-resource language.

5.3.2 Architecture

The architecture of this data collecting technology is shown in Figure 5.1. A particular speaker makes the recordings of a shown utterances using web application or mobile version of a website and then send these recording to our server with HTTP protocol. Such a way, it is possible to operate this application regardless of a current platform.

5.3.3 Technology

This application was developed with the latest available technologies that are suitable for any device or computer in the market today. The backend of an application was done using yii2-framework in php7.2 language. The architecture development for this project is based on Model-View-Controller (MVC) template.

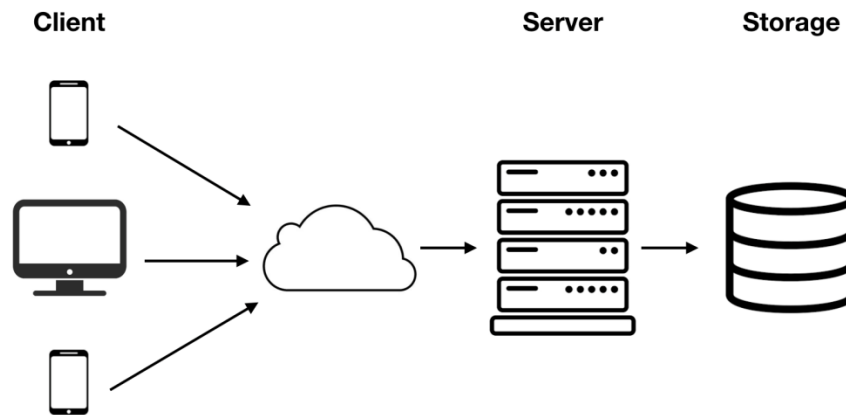


Figure 5.1 - The representation of system architecture

As a HTTP server we utilized an engine NGINX. For managing databases in the application, we have used the language based on MySQL. The control panel was handled by phpMyadmin and heidiSql. For code editing we utilized the phpstorm technology. Each web-page in applications has the ability to change the resolution based on the current device.

5.4 Speech synthesis of Kazakh language application

In order to automate the process of data collection, the speech synthesis using convolutional neural networks have been applied. The model generates the recordings that are only 3 seconds long and sentences with maximum 20 symbols. This approach makes the data collection technique faster and convenient in terms of spending time to short sentences.

The chapter describes the process of modeling a speech synthesis system based on available acoustic data used for machine learning in order to obtain a model that corresponds to the natural characteristics of speech. The characteristic features and architectures of CNN convolutional neural networks are presented, in which data is implemented as a set of images, and local convolution operations are used that modify and combine map image data with each other. Current methods of machine learning and the use of a GPU-accelerated library for deep neural networks are considered.

5.4.1 Speech synthesis

Usually, to build a speech synthesis system, you need a whole team of specialists from different fields. For each of them there is a whole host of algorithms and approaches. Let us start with a superficial understanding of each of them.

5.4.1.1 Linguistics

Normalization of the text. First, All abbreviations, numbers and dates need to be expanded into text. *The 50s of the XX century* should turn into *the fifties of the twentieth century*, and *the city of Almaty, Al-Farabi Ave.* into *the city of Almaty, Al-Farabi Avenue*. This should happen as naturally as if a person were asked to read what was written.

The preparation of stress dictionary. The accents can be arranged according to the rules of the language. In English, the emphasis often placed on the first syllable, and in Spanish - on the penultimate. Moreover, from these rules, there are whole hosts of exceptions that are not amenable to any general rule. They must be taken into account. For the Russian language in the general sense, the rules for the placement of stress do not exist at all, so without a dictionary, stresses cannot be placed. In addition, in the case of the Kazakh language, the problem of stress placement solved very simply. A pleasant specificity of the Kazakh language is that the main emphasis placed on the last syllable.

Removal of homography. Homographs are words that coincide in spelling but differ in pronunciation. Completely remove the homography without taking into account the context is impossible.

5.4.1.2 Prosodic

Syntagma highlighting and pause. A syntagma represents a relatively finished segment of speech in meaning. When a person speaks, he usually inserts pauses between sentences. We need to learn how to divide the text into such syntagmas.

Determination of the type of intonation. The expression of completeness, question, and exclamation is the simplest intonation. However, expressing irony, doubt or enthusiasm is a much more difficult task.

5.4.1.3 Phonetics

Getting transcription. Since the work is done with pronunciation, and not with writing, it is obvious that instead of letters (graphemes), it is logical to use sounds (phonemes). Converting a grapheme recording into a phoneme is a separate task, consisting of many rules and exceptions.

Calculation of intonation parameters. At this point, you need to decide how the pitch and the pronunciation speed will change. Depending on the placed pauses, the selected

sequence of phonemes and the type of expressed intonation. In addition to the basic tone and speed, you can experiment for a long time with other parameters.

5.4.1.4 Acoustics

Selection of sound elements. Synthesis systems operate with the so-called allophones - realizations of the phoneme, depending on the environment. Records from the training data cut into pieces by phoneme marking, which forms an allophone base. Each allophone characterized by a set of parameters, such as context (phonemes neighbors), pitch, duration, and others. The synthesis process itself is the selection of the correct sequence of allophones, the most suitable in the current conditions.

Modification and sound effects. For the resulting recordings, post-processing needs some special filters that make the synthesized speech a little closer to human speech or correct some kind of defects.

5.4.2 Implementations

The deep Convolved Text-To-Speech (TTS) from Kyubyong Park was chosen as the basis for future experiments. Let us take a closer look at the implementation.

The author laid out the results of the synthesis on three different databases and at different stages of training. The latest database in English (Kate Winslet's Audiobook) it contains only 5 hours of speech, which is also a great advantage for us since our database contains approximately a comparable amount of data.

Some time after that, the repository reported that the author had successfully trained the model for the Korean language. This is also quite important, since languages can vary greatly and robustness in relation to the language is a nice addition. You can expect that the learning process will not require a special approach to each set of training data: language, voice, or some other characteristics.

5.4.2.1 Initialization of main parameters

We want to create a network that accepts text as input and synthesizes sound as output. The abundance of implementations shows that this is possible, but there are of course a number of reservations.

The main hyperparameters of the system are put in a separate file, which named accordingly: `hparams.py` or `hyperparams.py`. Hyperparameters contain everything that can be changed without touching the main code. Starting from the directory for logs, ending with the size of hidden layers.

5.4.2.2 Text processing

For text processing, the so-called embedding layer usually used, which placed first. Its essence is simple — it is just a table that matches each character from the alphabet with a certain feature vector. During the training process, we select the optimal values for these vectors, and when we synthesize them from a ready-made model, we simply take the values from this very table. This approach used in the already well-known Word2Vec, where a vector representation for words built.

In the course of training, we found out that the optimal values of each of their characters are as in Appendix H.

Then for the *aabbcc* line after passing the embedding layer, we get the matrix $[[0,1], [0,1], [2,3], [2,3], [4,5], [4,5]]$.

This matrix further fed to other layers that no longer operate with the concept of symbol.

At this point, first restriction is observed: the set of characters that we can send to synthesis is limited. For each character, there must be a non-zero number of examples in the training data, preferably with a different context.

The alphabet of the Kazakh language has several important points and assumptions:

1. Alphabet punctuation was inserted. On the one hand, we really do not say them. On the other hand, we use punctuation marks to divide a phrase into parts (syntagmas), separating them with pauses.

2. in the alphabet, no numbers. The numbers expected to be converted to numerals before submitting for synthesis, i.e. normalized. In General, all E2E architectures require normalized text.

3. There are no Latin characters in the alphabet. The system will not be able to pronounce English.

4. There is an «*ë*» in the alphabet. In the data, in which system was trained, it stood where it needed to, and alignment decided not to be changed. However, when results were evaluated, it turned out that now before submitting for synthesis, this letter also needs to put correctly, otherwise the system pronounces «*e*», and not «*ë*».

5.4.2.3 Sound

Almost all systems operate not with the signal itself, but with various types of spectra obtained on Windows with a certain step. Let us focus on implementation and usage. The DCTS implementation uses two types of spectra: the amplitude spectrum and the Mel spectrum. It is shown in Appendix I.

Now let us see (Figure 5.2) how the amplitude spectrum looks on one of the files from the used database:

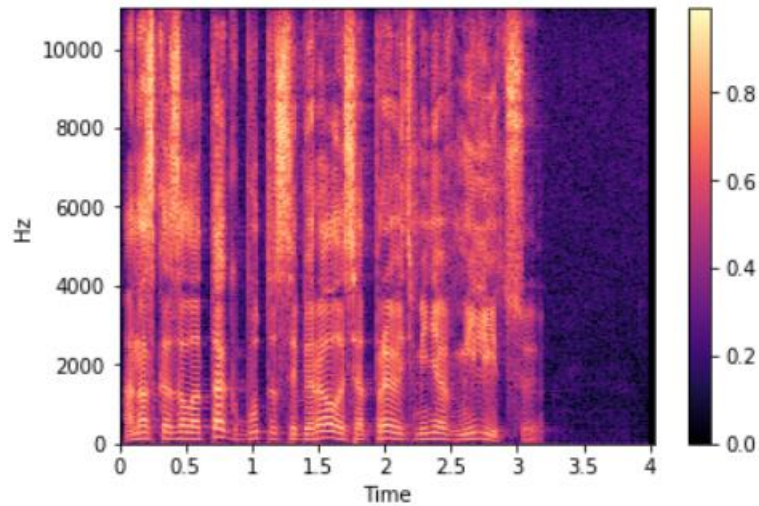


Figure 5.2 - The magnitude spectrum

This representation of window spectra called a spectrogram. On the x-axis is time in seconds, on the ordinate — frequency in Hertz. The color highlights the amplitude of the spectrum. The brighter the point, the greater the amplitude value.

A Mel spectrum is an amplitude spectrum, but taken on a Mel scale with a certain pitch and window. We set the number of steps in advance. In most implementations, the value 80 used for synthesis (set by the *hp.n_mels* parameter). Switching to the Mel spectrum allows you to significantly reduce the amount of data, but still preserve important characteristics for the speech signal. The Mel spectrogram for the same file looks like this (Figure 5.3):

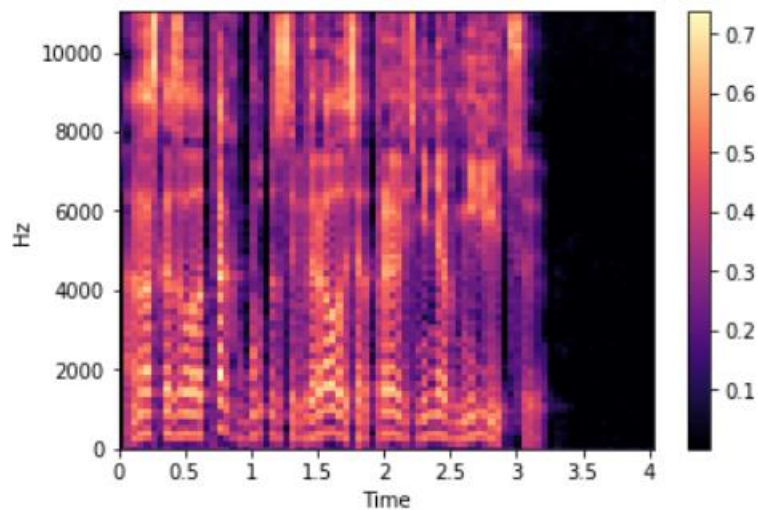


Figure 5.3 - The mel spectrum

Note the thinning of the Mel spectra over time on the last line of the listing. We take only every 4 vector ($hp.r == 4$), thereby reducing the sample rate accordingly. Speech synthesis reduced to predicting mel spectra from a sequence of characters. The idea is simple: the less the network has to predict, the better it will do.

Well, we can get a spectrogram from the sound, but we cannot listen to it. Accordingly, we need to be able to restore the signal back. For these purposes, systems often use the Griffin-Lim algorithm and its more modern interpretations (for example, RTISILA). The algorithm allows you to restore the signal by its amplitude spectrum. The implementation that we used is shown in Appendix J.

5.4.3 Neural network training

The DCTTS we chose consists of two virtually independent neural networks: the Text2Mel and Spectrogram Super-resolution Network (SSRN) (Figure 5.4).

Text2Mel predicts the Mel spectrum across text using an attention mechanism that links two encoders (TextEnc, AudioEnc) and one decoder (AudioDec). Note that Text2Mel recovers exactly the sparse Mel spectrum.

SSRN restores the full amplitude spectrum from the Mel spectrum, taking into account frame skips and restoring the sample rate.

We took a voice that includes 8 hours of recordings (several thousand files). Left only records that:

1. The text contain only letters, spaces, commas, periods.
2. The text Length does not exceed $hp.max_N$.
3. The length of the mel spectra after cutting does not exceed $hp.max_T$.

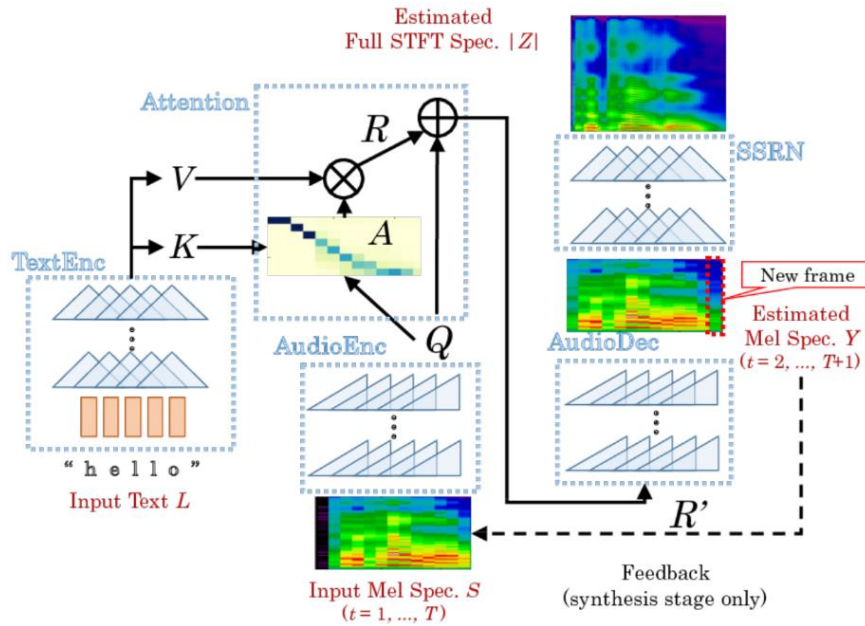


Figure 5.4 - DCTTS architecture

The result was more than 5 hours of recording. We calculated the required spectrograms for all records and started Text2Mel and SSRN training in turn.

DCTTS contains only convolutional layers, and unlike RN implementations like Tacotron, it learns much faster.

On our machine with Intel Core i5-4670, 16 GB RAM and GeForce 1080 on Board, 50 thousand steps for Text2Mel learn in 15 hours, and 75 thousand steps for SSRN-in 5 hours. The time required for a thousand steps in the process of learning from me has not changed much, so you can easily estimate how much time it will take to learn with a large number of steps.

The size of the Butch can adjust by the $hp.B$ parameter. Periodically, my learning process failed with out-of-memory, so I just divided by 2 the size of the Butch and restarted the training from scratch. I believe that the problem lies somewhere in the depths of TensorFlow and the subtleties of implementing batching.

Currently, speech synthesis is used in a variety of areas: voice assistants, information and reference systems, military and space technology, robotics, acoustic dialogue between a person and a computer. Until recently, speech synthesis systems were multi-module complex applications that implement a variety of algorithms and approaches. Modern speech synthesis systems are based on the use of neural network programming and machine learning methods, which have managed to automate the operation of most components of such systems.

Machine learning is a branch of artificial intelligence that is characterized not by a direct solution to a problem, but by learning in the process of applying solutions to many similar problems. Machine learning is located at the intersection of mathematical

statistics, optimization methods, and classical mathematical disciplines, but it also has its own specifics related to information extraction, data mining, and computational efficiency problems.

The scope of application of machine learning technologies is wide and is constantly increasing, including image recognition, medical and technical diagnostics, statistical analysis, forecasting, management and decision-making tasks, processing text arrays, and speech synthesis. For example, in the field of linguistics, machine learning helps determine the grammatical characteristics of words; in phonetics, machine learning is used to predict the frequency of the main tone and the length of phonemes. Prosodic characteristics, such as determining the length and location of pauses, and predicting intonation, can also be trained. This chapter describes the process of modeling a statistical speech synthesis system based on available acoustic data used for machine learning of the system in order to obtain a model for matching speech characteristics. Machine learning is based on data, and the effectiveness of speech recognition and synthesis depends directly on their quality and quantity.

Therefore, a necessary condition for implementing machine learning methods is the availability of a large amount of data in the form of a specific language corpus based on the use of acoustic-phonetic, text, and speech databases.

5.4.4 CNN for speech synthesis

The most common machine learning models are artificial neural networks, which used to process input data from a combination of distributed simple operations that depend on the parameters being trained. Modern neural networks used in forecasting, image and speech recognition, text generation, and many other tasks. The most optimal types are convolutional networks and recurrent neural networks.

To create the speech synthesis system described in this chapter, a deep convolutional neural network was used, which has been successfully used by other researchers in pattern recognition, audio signal processing, and speech processing. A characteristic feature of convolutional neural networks is the use of data representation in the form of a set of images (maps) and the use of local operations-convolutions that modify and combine map data with each other.

Figure 5.5 shows a model of a convolutional neural network consisting of only two 2D convolutional layers, followed by the global maximum pool layer (GlobalMaxPool). The model input is represented by a feature matrix X of size $N \times M$. The first convolutional layer consists of K filters of $1 \times M$ size with the ReLu activation function. The second convolutional layer has only 1 filter size 1×1 . It combines the functions selected by the convolution filters on the previous layer for each of the N candidates separately. This layer does not have an activation function. After that, the GlobalMaxPool layer selects the candidate with the maximum activation, thus assuming that the candidate contains a harmonic as a component of the input signal. Then the selected candidate goes through

the sigmoid activation function, as a result, the network output will be represented by the probability that the input signal is a voice.

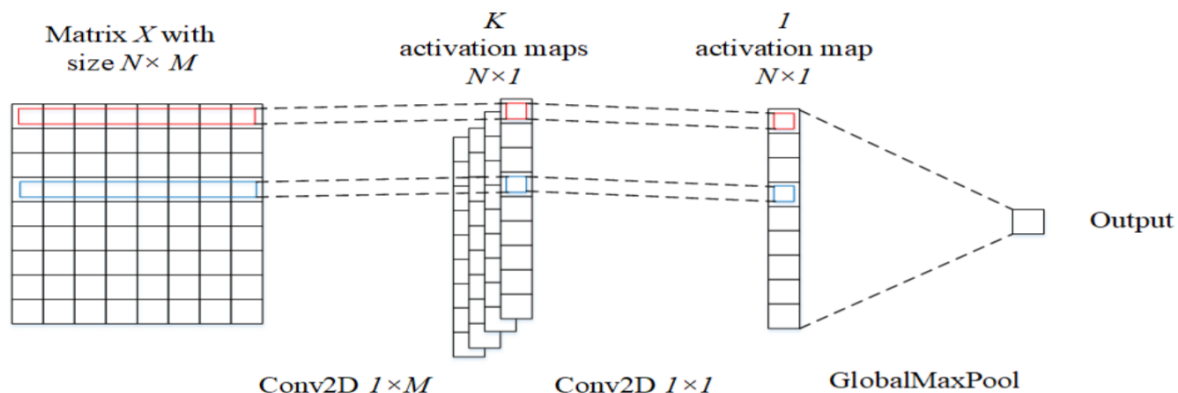


Figure 5.5 - Architecture of a 2-dimensional convolutional network

The efficiency and quality of a neural network depends on the choice of its architecture. When modeling speech synthesis systems, deep convolutional neural networks are often used, which consist of 1000 to 2000 neurons in each of the 6-8 hidden Layers (Figure 5.6).

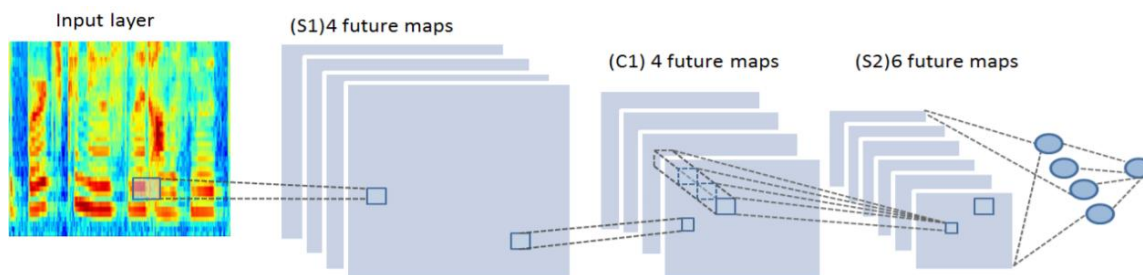


Figure 5.6 - Architecture of a multidimensional CNN network

Preparation of training data The basis of the methodology for creating speech databases of natural languages used for machine learning is the presence of a large volume of recorded speech (from 15 hours), in addition, the corpus must be balanced and as complete as possible, that is, contain all the necessary units in all possible contexts with different characteristics. Previously, an experimental training database (5 hours of speech) was compiled, and audio materials that are freely available were used.

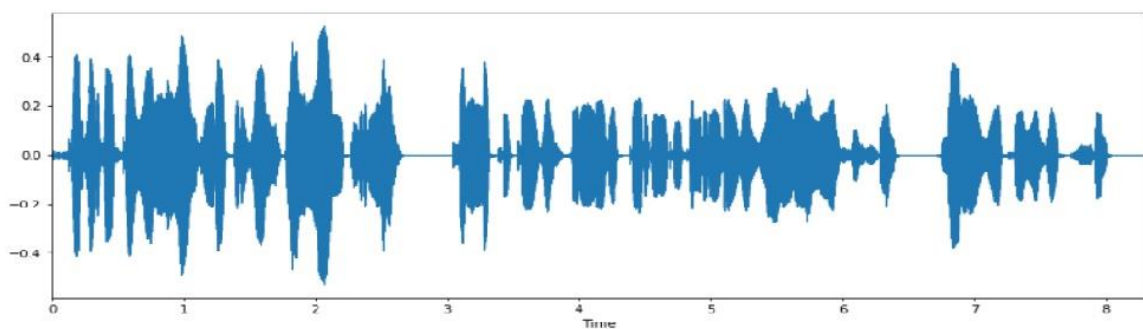
When building speech synthesis systems, one of the most important tasks is to segment and label databases of speech signals into semantically and phonetically significant units of speech, and in our particular case, these are phonemes. The resulting

segments are stored in the database and used for machine learning of acoustic models in the integrated system, with subsequent generation of the speaker's voice in the text-to-speech synthesis system. One of the specific methods of working with trained systems is to set the system parameters for a specific language and select the alphabet. Since the Kazakh speech synthesizer model uses phonemes as input alphabet symbols, it was necessary to solve the problem of transcribing texts according to the rules of grapheme to phoneme, taking into account the phonetic features of the Kazakh language. The task was solved by creating a phonetic transcription module, and then the prepared text-training sample was transcribed.

Thus, a training experimental base was created, consisting of 3500 sentences, and each sentence corresponds to an audio file in wav format with a sampling frequency of 22050 Hz. After that, the system module was activated, which is responsible for receiving spectrograms of audio files, on the basis of which deep learning of networks takes place (Figure 5.7).

Machine learning of the system was based on libraries and modules of the Python programming language (Tensorflow, matplotlib, numpy, scipy).

The main training stage of the speech synthesis system consists of two networks: Text²Mel, which synthesizes the Mel spectrogram from the input text, and Spectrogram Super-resolution Network (SSRN), which converts the Mel spectrogram of the signal to the STFT amplitude spectrogram, taking into account frame skips and restoring the sample rate.



білім беруде, денсаулық сақтау мен көлік саласында болады.

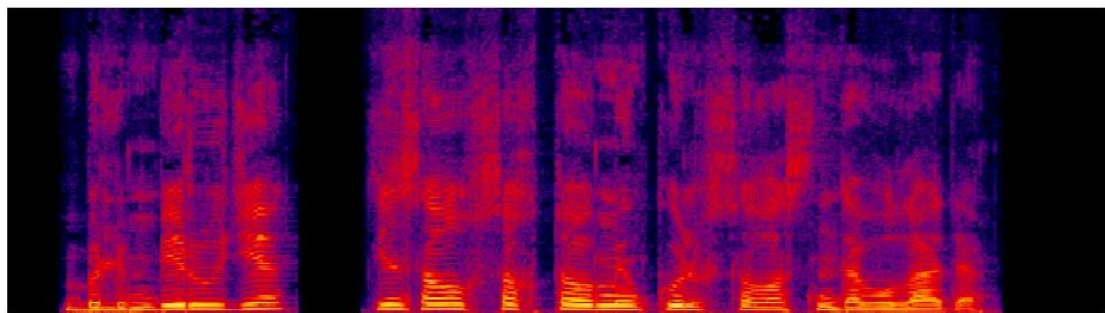


Figure 5.7 - Sample offer from the training database

5.4.5 Description of the machine learning process

Machine learning is implemented as a process of computer analysis of prepared statistical data in order to search for patterns and create the necessary algorithms based on them, configure the parameters of the neural network and further predict the operation of the system.

There are 3 main ways of machine learning:

- training with a teacher;
- training with reinforcement;
- training without a teacher (self-study).

In training with a teacher, specially selected data is used, in which the correct answers are already known and reliably determined, and the parameters of the neural network are adjusted so as to minimize error. In this method, the artificial intelligence system can match the correct responses to each input example and identify possible dependencies of the response on the input data. You can train the system on very large amounts of input data and vary the parameters of neural networks until the output results are expected. After that, you need to check how the system predicts the result for control data that the computer has not yet encountered.

Most often, training with a teacher is used to solve classification, regression, and forecasting problems. For example, there are classification algorithms for analyzing text

and determining parts of speech of a particular language in texts of various subjects. regression can be used to categorize text documents.

Reinforcement learning is based on regulated learning processes in which the machine learning algorithm is provided with a set of actions, parameters, and end values. After defining the rules, the machine learning algorithm tries to explore different options and possibilities, tracking and evaluating the result each time to determine which of the options is optimal. Supported learning is a trial-and-error method for a machine. She learns from past experience and changes her approach, reacting to a new situation, trying to achieve the best possible result.

If you use a method of teaching without a teacher, the machine learning algorithm examines the data in order to identify patterns. There is no ready-made data with responses or an operator that can train the machine. On the contrary, the program itself determines correlations and relationships based on the analysis of available data. When teaching without a teacher, the machine learning algorithm is allowed to independently interpret large data sets and draw conclusions based on them. The algorithm tries to sort the data in some way and describe its structure. This may look like grouping data into clusters, or it may look like ordering data so that it starts to look organized.

As data becomes available for analysis, the ability of the algorithm to make decisions based on this data, as well as the accuracy of these decisions, increases. A good example of self-learning is clustering, which involves grouping similar data based on certain parameters. This can be used to segment data into multiple groups and perform analysis based on each data set separately to find patterns. Learning without a teacher can also be used to analyze the tone of utterances to determine the emotional state of people based on their spoken and written speech.

Deep learning mechanisms typically use multi-layer neural networks and a very large number of object instances to train the neural network. The number of records in a training sample can number hundreds of thousands or even millions of examples, and when resources are not limited – even more.

The size of the training database and the number of input data entries depend on how the machine creates the necessary classification rules. The more heterogeneous data is uploaded to the system at the machine learning stage, the more accurately these rules will be detected, and the more accurate the result of the speech synthesis system will be in the end, and the speech signal generated at the output will be of higher quality.

For novice researchers in the field of machine learning and its application for speech synthesis, it is advisable to use machine learning methods with a teacher. This software requires less time and financial resources when creating a prototype of a working system and practical development of artificial neural network techniques. In this case, you can get a functioning artificial intelligence system for a specific task faster. Currently, there are a large number of freely available high-quality libraries of software code for artificial neural networks, such as TensorFlow for mathematical modeling, OpenCV for image recognition problems.

Speech generation with a pre-trained model can be performed on the GPU and CPU, but the speed of speech synthesis on the CPU in our case has decreased tenfold.

The training of the system lasted 27 hours, 500 thousand iterations were performed for the Text²Mel neural network, and 270 thousand iterations were performed for the SSRN network.

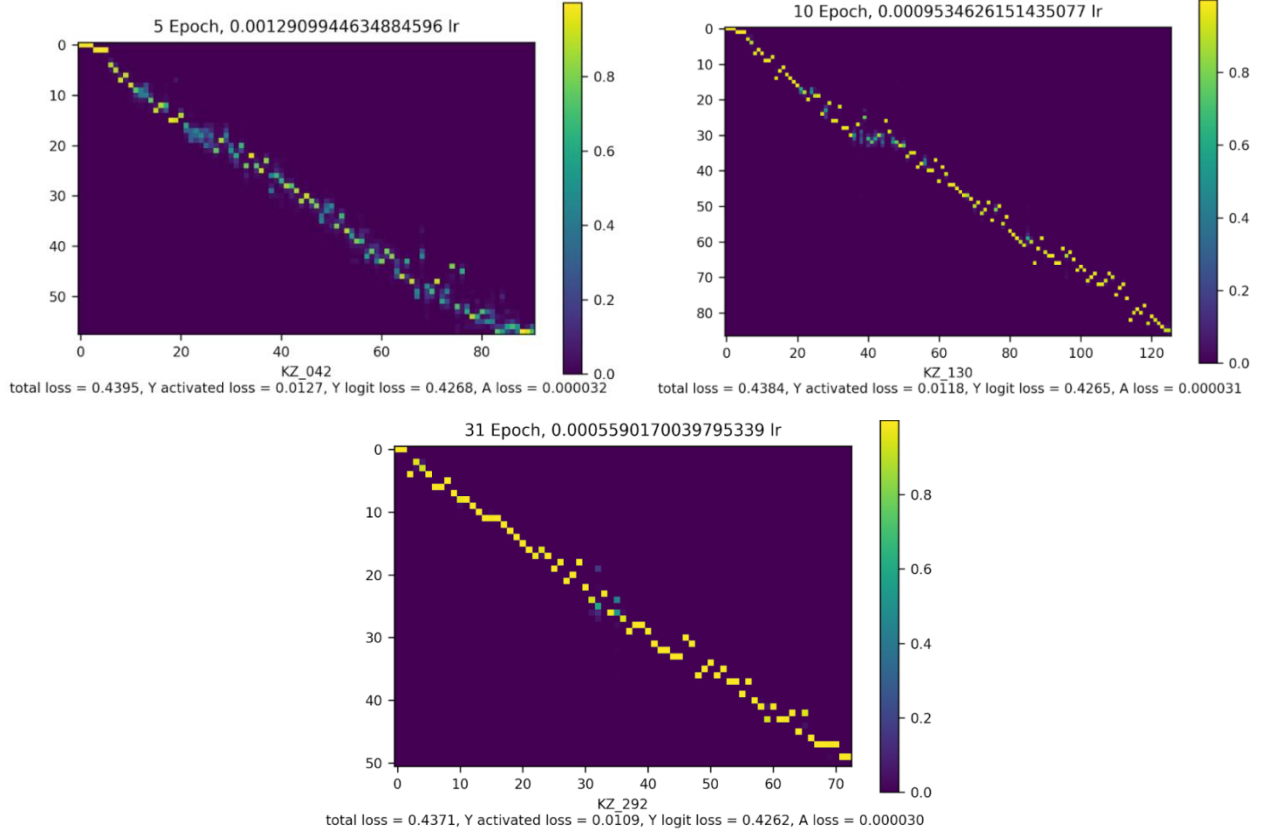


Figure 5.8 - The dynamics of learning matrix attention

Figure 5.8 shows the dynamics of improving the indicators of the attention matrix when training a neural network (50 thousand iterations, 100 thousand iterations, 310 thousand iterations).

The TextEnc network first encodes an input sentence $L = [l_1, \dots, l_N] \in \text{Char}^N$ consisting of N characters in two matrices $K, V \in \mathbb{R}^{d \times N}$. On the other hand, the AudioEnc network encodes a large-scale spectrogram $S(= S_{1:F, 1:T}) \in \mathbb{R}^{F \times T}$, of previously recorded speech, whose length is T , into a matrix $Q \in \mathbb{R}^{d \times T}$.

$$(K, V) = \text{TextEnc}(L) \quad (5.1)$$

$$Q = \text{AudioEnc}(S_{1:F, 1:T}) \quad (5.2)$$

The attention matrix $A \in \mathbb{R}^{N \times T}$ defined as follows evaluates how strongly the n -th character l_n and the t -th time interval are related

$$A = \text{softmax}_{n\text{-axis}}(K^T Q / \sqrt{d}) \quad (5.3)$$

$A_{nt} \sim 1$ implies that the module looks at the n -th character l_n on the time interval t , and it will look at the l_n or l_{n+1} or the characters around them on the subsequent time interval $t + 1$. Whatever happens, you should expect them to be encoded in the n -th column V . Thus, the initial number $\in \mathbb{R}^{d \times T}$ decoded in subsequent frames $S_{1:F, 2:T+1}$, is obtained as:

$$R = \text{Att}(Q, K, V) := V A. \quad (5.4)$$

The result R is combined with the encoded sound Q , as $R' = [R, Q]$. The concatenated matrix $R' \in \mathbb{R}^{2d \times T}$ is then decoded by the Audio Decoder module to synthesize a large scale spectrogram:

$$Y_{1:F, 2:T+1} = \text{AudioDec}(\hat{R}) \quad (5.5)$$

At the stage of synthesis of the matrix of attention and sometimes may not be able to read some symbols. Typical errors that occur most often are:

- sometimes several characters are skipped;
- repeatedly reads the same word twice or more.

To make the system more reliable, it is necessary to heuristically modify the a matrix as «almost diagonal». After this, it is observed that the device sometimes begins to mitigate such errors

5.5 The instrument

The system consists of two important roles to complete the whole process of data collection. It is an admin user (only 1-2 users) and large number of clients. Admin is eligible to add any number of books in any format, delete and edit. Admin can view the completeness percentage of a particular book and can listen to all recording by clients. He can list the recordings of any book as well as any speaker. He has the ability to delete the unwanted or corrupted recordings and email the speakers who make the mistake. After the whole recording process is finished the admin can download all records. The downloaded zip file already structured with folders of speakers and all necessary transcription with corresponded audio files.

The client can only register to a website by entering name, gender and age and choose any book listed by admin for further recording process. The client has no eligibility to add, delete or edit any book except listening to his own recordings for an improvement purposes. In the recording process client has the ability to rerecord the current sentence after listening his record repeatedly.

In the following sections we will discuss the detailed steps of the whole process with corresponding illustrations. The sections consist of two parts (admin and client).

5.5.1 Admin's part

As it was mentioned above admin user controls all the necessary parts of the collection process starting with adding book and ending with deleting unwanted records. Besides that admin has the ability to observe all the activities related data collection. It is the numbers of users, number of added books, number of completed books, number of all sentences, number of recorded sentences and overall memory taken by these recordings and sentences (Figure 5.9).

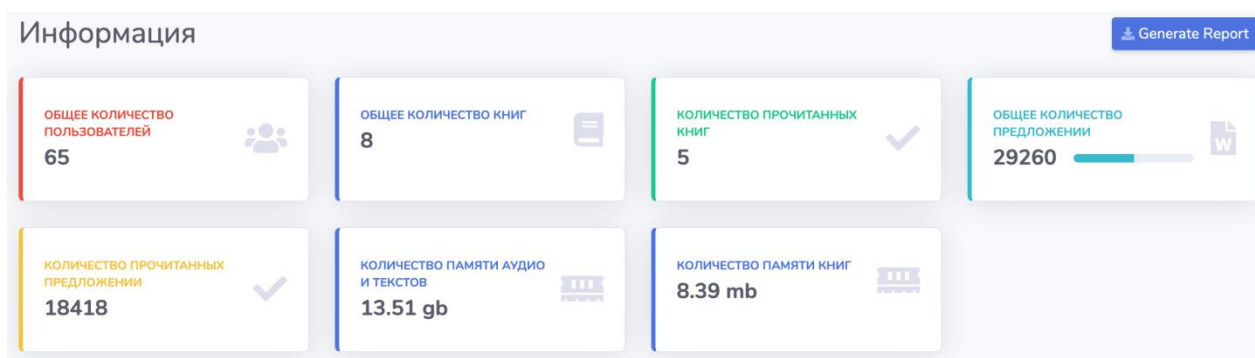


Figure 5.9 - Activity indicators

Since the memory of the server is limited it is important giving the admin ability to observe the statistics and indicators in order to exclude some unexpected circumstances. Moreover, admin just as a client has the ability to make the recordings of his own.

5.5.1.1 Adding book

The first important step of the whole process is the addition of the books. The illustration of this process is shown in Figure 5.10. There are three fields that admin user should fill before pressing save button. They are name of the book, naming structure of the files and format of an audio. The naming can be structured by username, directory name and create time name. The format of an audio file can be either mp4 or wav. Usually, researchers choose the wav format since the feature extraction from this format can be more valuable. Admin can choose any book in any language in any preferred format. Once

the book is added all sentence from current book gets split, organized and sorted by their length. Too long or too short sentences will be deleted. The unnecessary symbols and punctuations will be removed from the sentences automatically.

Добавить книгу

Название книги (Наименование папки)

Структура наименования файлов

Формат аудио

Добавить файлы...

Начать загрузку

Отменить загрузку

Удалить

☐

Save

Figure 5.10 - The book addition

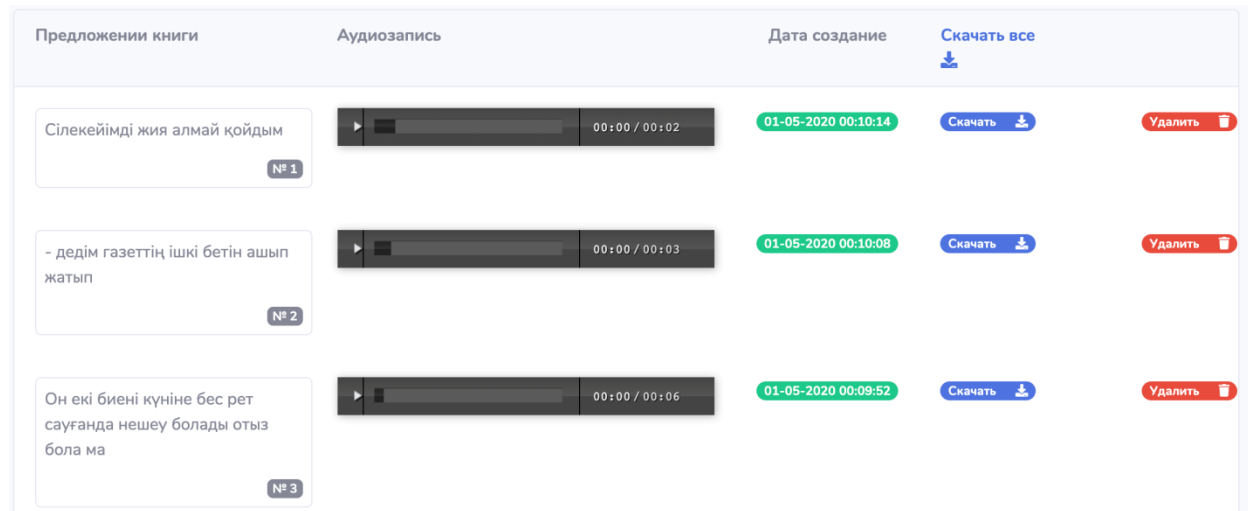
5.5.1.2 List of all books

After the book addition process the admin user can view all the added books with their names, id number, completeness percentage and creation date as it is shown in Figure 5.11.

#	id	Название книги (Наименование папки)	Процент прочтение книги	Дата создания	Дата изменения	
	<input type="text"/>	<input type="text"/>				
1	31	Соғыс психологиясы	<div><div>100%</div></div>	17 мар. 2020 г.	27 апр. 2020 г.	
2	32	Алпамыс батыр	<div><div>100%</div></div>	17 мар. 2020 г.	25 апр. 2020 г.	
3	33	Менің атым Қожа	<div><div>100%</div></div>	17 мар. 2020 г.	29 апр. 2020 г.	
4	34	Робинзон Крузо	<div><div>100%</div></div>	17 мар. 2020 г.	30 апр. 2020 г.	
5	35	Ақиқат пен аңыз	<div><div>50%</div></div>	17 мар. 2020 г.	17 мар. 2020 г.	
6	36	Ұлпан	<div><div>46%</div></div>	14 апр. 2020 г.	14 апр. 2020 г.	
7	37	Мартин Иден	<div><div>100%</div></div>	14 апр. 2020 г.	30 апр. 2020 г.	
8	38	Махаббат, қызық мол жылдар	<div><div>40%</div></div>	14 апр. 2020 г.	14 апр. 2020 г.	

Figure 5.11 - List of books

Once the every single sentence of the particular book is recorded the percentage indicator reaches 100 percent. Admin user can delete the any book by pressing red trash icon right after downloading the book. If admin deletes a book, it also gets deleted from the server database freeing up the memory. In order to view all the recording of a particular book, admin can click the eye button near to the trash button. Clicking this button leads to a next page listing all recordings as shown in Figure 5.12.



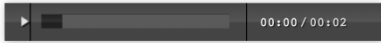


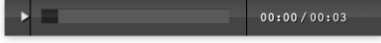


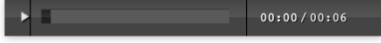


Предложения книги	Аудиозапись	Дата создание	Скачать все
Сілекейімді жия алмай қойдым № 1	 00:00 / 00:02	01-05-2020 00:10:14	Скачать  
- дедім газеттің ішкі бетін ашып жатып № 2	 00:00 / 00:03	01-05-2020 00:10:08	Скачать  
Он екі биені күніне бес рет сауғанда нешеу болады отыз бола ма № 3	 00:00 / 00:06	01-05-2020 00:09:52	Скачать  

Figure 5.12 - List recordings

Once the admin lists the recordings of any book, he can perform all kinds of operations. User can view the transcription itself and the corresponding audio recording. Next to the audio there is information about the recording (speaker name, creation date and time). On the right corner there are two buttons, which download button and delete button. Admin user checks and validates a specific recording listening to it by clicking the play triangular play button. As he listens to the recording depending on the quality and correctness the audio either gets deleted or stayed. The audio recordings are considered to be corrupted if there is an unwanted background noise presented or the audio itself doesn't match with corresponding transcription on the left. Once the audio gets deleted by pressing red delete button it cannot be restored from the server because of the memory limit issues.

Admin can also download the whole book by pressing download all button right on top. Once the downloaded zip file gets extracted transcription text files and recorded audio files will be structured as in Figure 5.13. If admin user wants to download a specific sentence with audio separately he can press the download button right on the list next to the delete button. If the admin wants to notify the speaker related the bad quality of recordings or corrupted audios he can easily email the speaker directly.

Name	^	Date Modified	Size	Kind
0_Farukh Iskalinov.txt		4/24/20	70 bytes	Plain Text Document
0_Farukh Iskalinov.wav		4/24/20	352 KB	Waveform audio
1_Farukh Iskalinov.txt		4/25/20	13...ytes	Plain Text Document
1_Farukh Iskalinov.wav		4/25/20	639 KB	Waveform audio
2_Baimolda Aray.txt		4/13/20	11...ytes	Plain Text Document
2_Baimolda Aray.wav		4/13/20	524 KB	Waveform audio
3_Farukh Iskalinov.txt		4/25/20	12...ytes	Plain Text Document
3_Farukh Iskalinov.wav		4/25/20	606 KB	Waveform audio
4_Farukh Iskalinov.txt		4/25/20	44 bytes	Plain Text Document
4_Farukh Iskalinov.wav		4/25/20	336 KB	Waveform audio

Figure 5.13 - Files structure

5.5.1.3 List of users

The list of users can be viewed by simply clicking “users” button in Figure 16. Once the page is opened it will display all registered users with their corresponding information. The admin user can observe user’s id numbers, their name, gender, age, number of recordings and registered date. Admin can search for the user by entering his id number or his first name and last name. The illustration of this page is demonstrated in Figure 5.14.

#	id	имя пользователя	статус	роль	Пол	Возраст	Кол-во прочитанных предложений	дата создание	дата изменение	
	<input type="text"/>	<input type="text"/>	выберит							
1	95	Bexultan	активный	student	Мужчина	19 лет	0	30 апр. 2020 г.	30 апр. 2020 г.	
2	94	Maratuly Ali	активный	student	Мужчина	19 лет	515	30 апр. 2020 г.	30 апр. 2020 г.	
3	93	Azamat00	активный	student	Мужчина	19 лет	398	29 апр. 2020 г.	29 апр. 2020 г.	
4	92	Madi Bikamalov	активный	student	Мужчина	20 лет	505	29 апр. 2020 г.	29 апр. 2020 г.	
5	91	abai	активный	student	Мужчина	18 лет	0	29 апр. 2020 г.	29 апр. 2020 г.	
6	90	Fotimuah	активный	student	Женщина	15 лет	1	29 апр. 2020 г.	29 апр. 2020 г.	

Figure 5.14 - List of users

Moreover, admin can observe the user’s current status. If the particular user hadn’t been recording for the last 10 days, he is considered to be inactive. Otherwise, the speaker will be highlighted as an active user.

Admin can also view the recordings of a specific user by click eye button on right side of the list. It will take to the page as it was described in Figure 19, but this time it will display the recordings of the speaker. This gives more convenience on controlling the quality of the data collection process, since admin can directly email the speaker mentioning his mistakes and errors.

Not only admin has the eligibility to control an entire process, he can also influence on it in terms of recording his own audios. This can be done by clicking the “View book” button in Figure 5.11.

5.5.2 Client’s part

The client or speaker doesn’t have much privilege compared to admin user. The only concern of the speaker is the quality of his own recordings. He cannot influence the process by adding, deleting or editing the books. The speaker can only see the currently available books (with completeness indication) and record them. Other than that, the client has no eligibility on controlling the data collection operation.

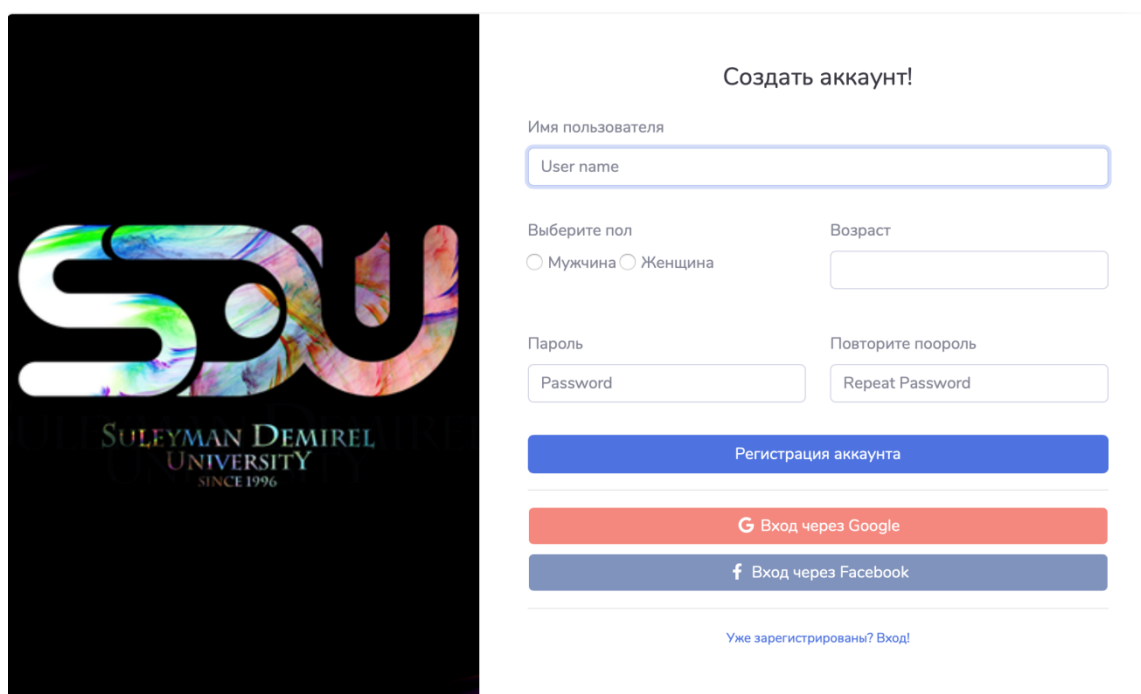
The image shows a user registration interface. On the left is a black banner with a colorful, abstract logo that looks like the letters 'SDU' in a stylized, multi-colored font. Below the logo, the text 'SULEYMAN DEMIREL UNIVERSITY' is written in a serif font, with 'SINCE 1996' in a smaller font below it. On the right is a white registration form titled 'Создать аккаунт!'. The form contains several input fields: 'Имя пользователя' (User name) with a text box labeled 'User name'; 'Выберите пол' (Select gender) with radio buttons for 'Мужчина' (Male) and 'Женщина' (Female); 'Возраст' (Age) with a text box; 'Пароль' (Password) with a text box labeled 'Password'; and 'Повторите поороль' (Repeat password) with a text box labeled 'Repeat Password'. Below these fields are three buttons: a blue button labeled 'Регистрация аккаунта' (Register account), a red button labeled 'G Вход через Google' (Sign in with Google), and a blue button labeled 'f Вход через Facebook' (Sign in with Facebook). At the bottom of the form, there is a link that says 'Уже зарегистрированы? Вход!' (Already registered? Sign in!).

Figure 5.15 - User registration

The first step for the user is to go through the registration phase as illustrated in Figure 5.15. The speaker accomplishes it by entering his name, age, gender and password. This information will be further used and analyzed to make the statistics on users.

Once the registration process is done, speaker will be redirected to the main page containing the list of books with all necessary information (name of the book, id number,

completeness percentage). On top of this list speaker can observe his number of completed recordings as it's shown in Figure 5.16.

SDU AudioRecording

Панель управлениеВыход

Вы прочитали 52 предложения.

Показаны записи 1-2 из 2.

#	id	Название книги (Наименование папки)	Процент прочтение книги	Дата создания	
	<input type="text"/>	<input type="text"/>			
1	26	Алпамыс батыр	5% <div></div>	14 мар. 2020 г.	Начать чтение
2	30	Соғыс психологиясы	0%	16 мар. 2020 г.	Начать чтение

Figure 5.16 - List of books for speaker

On the left side of the list, there is a “Start reading” button, which leads to the main idea of the whole process. All sentences of the books will be split and randomized to each user. This will help to avoid some collisions between users (two users recording the same sentence). We understand that this will not entirely remove this problem, but at least it will reduce its possibility.

After clicking the “Start reading” button, it will open the giant page with the first sentence displayed on it. The illustrated view of this page is showed in Figure 5.17. On the top there is an instruction that explains the recording process in detail. On top right side of the page speaker can observe the number of sentences left to complete the book. Right below the sentence there is a “start recording” button. By clicking this button the website starts the recording process and user immediately should begin the pronouncing the utterance. Right after, the speaker finishes the recording operation he should press the same “start recording” button. Moreover, user can record the audio repeatedly by pressing this button again. This makes the recording process a lot convenient.

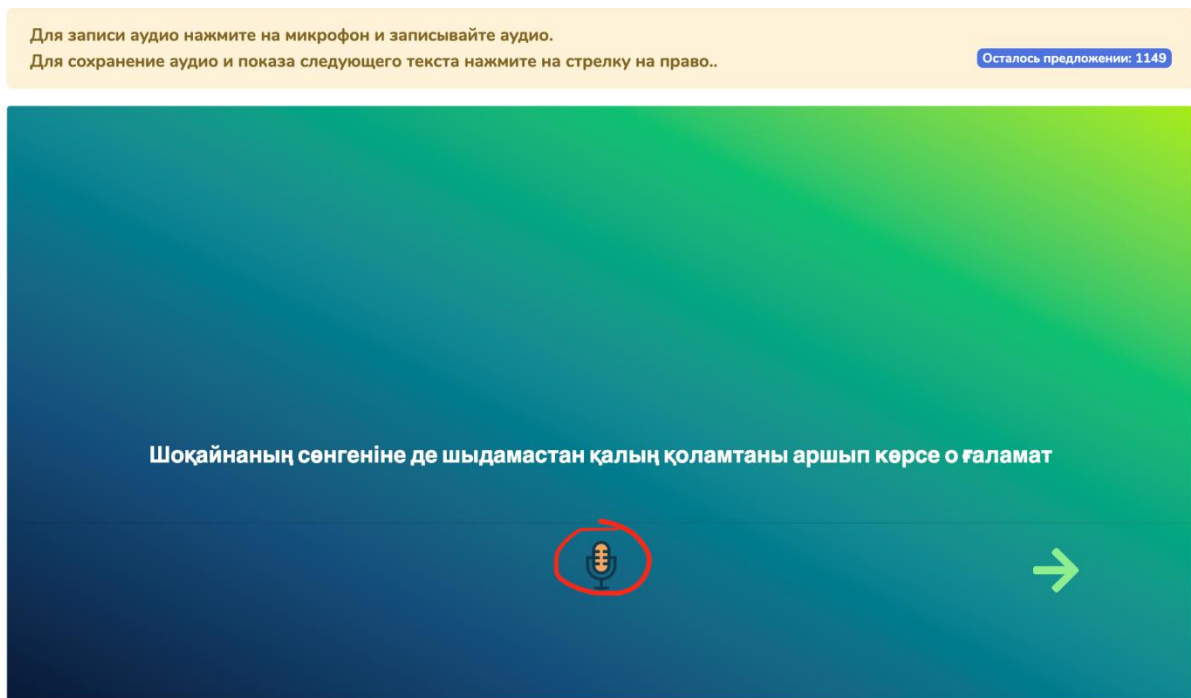


Figure 5.17 - Recording page

After the sentence is recorded right below the sentence there is a mini audio players get appeared. Speaker can listen back to the recorded audio by pressing a play button. If the recording is too slow or too fast or contains some noisy external sounds, user can record the sentence again. This gives an opportunity for the user to take control over the quality of collected data. Once user is convinced that the recording is suitable for a good quality, he can save the audio and jump into the next sentence by pressing the arrow button on the right. By HTTP request the audio file as mp4 or wav and transcription as txt is stored in the database. After that admin user can view the recorded sentence and evaluate the quality.

One more important possibility for the user is the freedom. He can record the sentence from one book and then go over to another book's sentence. This makes the collection part more interesting.

5.6 Evaluation

With this instrument we have collected over 20 hours of speech. Around 65 speakers were involved in this process, where 54 make and 11 female. The main audience was selected from universities. Each of the speakers on average contributed on operation recording around 300 sentences. In Figure 5.18 we can see the gender distribution of speakers. It is clear that there are far more male speeches than female speeches (83% to 17%). This may cause a bias, but with augmentation process this issue can be resolved.

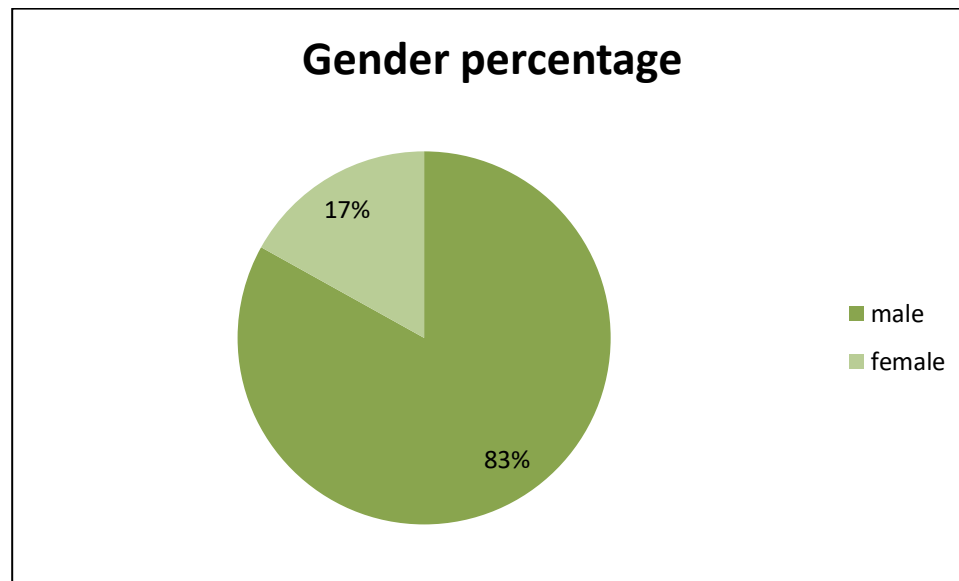


Figure 5.18 - User statistics over gender

In Figure 5.19 we can observe the age distribution of speakers who are over 20 and under 20 (11% to 89%). As we can see, the data samples are hugely biased towards the age under 20. This data can't be considered as a local error of an application itself, because most of the speakers were chosen from universities.

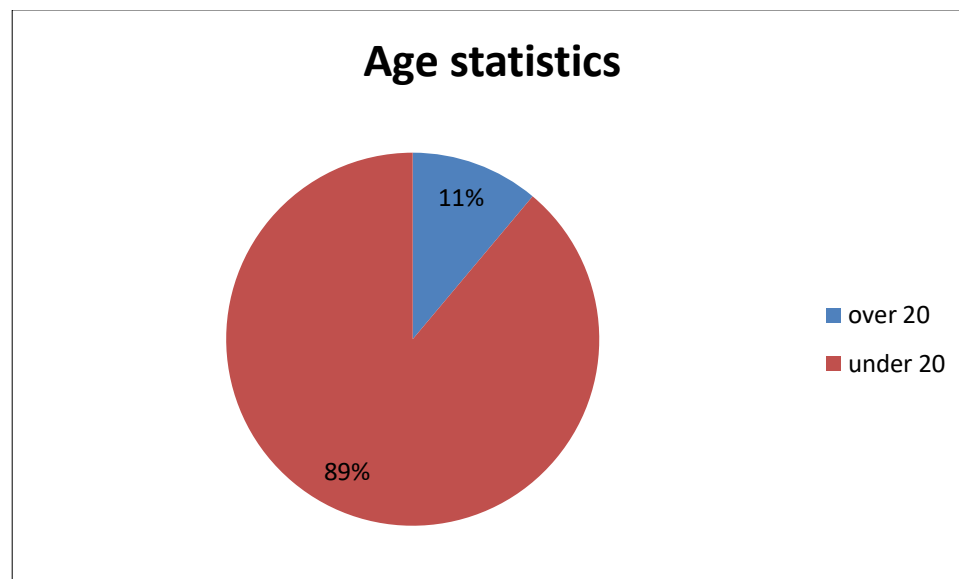


Figure 5.19 - User statistics over age

As a result, we have obtained 18411 of clean audio recordings with corresponding transcriptions. The average duration of each audio clip is around 4.5 seconds. The total amount of duration is 20 hours. The gathered data was manually checked and controlled by listening repeatedly and discarding unwanted and corrupted recordings. We have

discovered that error percentage of all recordings is 10%, since the overall number of audio clips reached 20441. In Table 5.1, we can observe the list of books, from which the sentences were extracted and the number of recordings of each book with an error percentage.

Table 5.1 - Recording numbers

Name	Clean recordings	Corrupted recordings	Error %
Соғыс психологиясы	2166	301	13.9%
Алпамыс батыр	1070	247	23%
Менің атым қожа	2823	178	6.3%
Робинзон Крузо	1703	251	14.7%
Ақиқат пен аңыз	2798	208	7.4%
Ұлпан	3293	99	3%
Мартин Иден	1699	289	17%
Маххабат, қызықмолжылдар	2859	457	16%

5.7 Discussion

The researchers who tend to construct their own speech corpus try to concentrate more to web-based methods a lot recently, because of the web application popularity. As we have discussed earlier, web-based crowd-sourcing approach is considered to be one of the most effective ways on building a speech corpora. A lot of the biggest companies related to speech recognition utilize their speech samples obtained from smartphone devices or web-applications to train their systems [115]. These big industries train their models on huge datasets (over 18 000 hour for Google Home tasks), which makes it almost impossible to outperform their model for an independent researcher.

If the data collection for ASR system is being executed for a specific domain or area, the collection process should be a little different. As it was described above, the collection of data using web searching is completely inapplicable to construct a corpus.

Considering above obstacles, the low resource languages like Kazakh have a difficulties in building large corpuses. Compared to other popular languages, such languages in the web-resources suffer from inexistence. Even if there are some sources where one can get access to data, it is most probably costs valuable amount of money, which can be unaffordable for an independent researcher. Introduction of our crowd-

sourcing instrument surely will increase the popularity in building ASR systems for low-resourced languages.

These days almost all people on the planet use smartphones with advanced microphones and computers. Therefore, we can take advantage of this fact by involving number of people to data collection process to build speech recognition system. However, it is completely inefficient to ask people record on their own smartphones without providing instructions and rules of this process. Moreover, the sentences and their quantity should be provided beforehand. In addition, the main problem is obtaining clean and clear audio speech recording in uncontrollable conditions, since speakers are completely on their own environment. All these issues can be solved by using our tool on speech data collection. As it was mentioned in above sections, there are very few instruments based on web or mobile for such tasks.

The instrument that we are proposing is well-organized and convenient application on web platform, which can be adapted to any smartphone with all necessary guidance to potential user. Therefore, any people can be involved in this technique since almost any individual uses smartphones or at least personal computers. The researchers can benefit from such technology, since it provides big amount of people and short time of data collection. Moreover, it provides good quality of audio speech recording, because of the ability to monitor and control the whole process of recordings. Right now, all the instructions and rules are presented in Kazakh language, but it can be adapted to any low-resourced language. Therefore, this tool will contribute huge effort on the popularity of poor famous languages by helping to collect the speeches in convenient and painless way.

5.8 Summary

There are a lot of speech datasets for popular languages on the internet that help researchers to make a proper experiment on ASR systems. However, this is not the case for low-resourced languages like Kazakh language. ASR systems for such languages poorly investigated and probably don't exist, because of the lack of data. There are many techniques and methods for researchers to create their own speech corpus for any languages. However, the all are poorly effective and nearly impossible to execute on low-resourced languages. All of these drawbacks we have explored and discussed in the section above. To avoid these drawbacks, we have built an instrument based on web application that helps to collect speech data. This website is also adapted to any smartphone screen, since the smartphones are our daily life device.

Usage of this technique helps on two main challenges: the good quality of speech recordings and huge size of a data. The good quality of audio clips is obtained because of the well-designed monitoring system and proper recording system. Huge size of a speech data is provided by users who can be easily involved in contributing the process.

We have collected 30 hours of speech recording with their corresponding transcriptions using our tool. There were 65 speakers' involved (54 males, 11 females).

Almost all of them are university students. Each of them on average recorded on average around 300-500 sentences. After all monitoring and checking process, deleting all unnecessary audio clips we have obtained 18411 recordings. This number is very promising, because the data collection process lasted only 1.5 month.

The technique that we utilized to construct speech corpus was completely successful and we are planning to conduct this experiment again in the future. We hope that this data collection technology will be helpful for any researcher who is suffering from lack of data for low-resourced languages.

6 IMPLEMENTATION OF ASR SYSTEM FOR KAZAKH LANGUAGE

The lack of speech data is very common problem for unpopular languages. Therefore, the investigation in ASR area is very poor, since the speech recognition models should be trained on huge amount of data. Unfortunately, Kazakh language is considered to be low-resourced language due to its unpopularity around the world. There are almost no existing speech corpuses with transcriptions on Kazakh language. Even if there are some manually collected corpuses, they are not enough for achieving worth mentioning result and moreover, they are not available for free.

To implement the ASR system for Kazakh language, we have first collected using our above mentioned instrument 30 hours of speech data. In addition, we have pre-existing 20 hours speech corpus based on Kazakh language as well. It was obtained by manually collecting from Kazakh audio books and by web-sourcing through the internet. To completely erase the problem of lack of speech data for Kazakh language, we have used the pre-trained model based on Russian language. It was trained using recurrent neural networks on Russian 100 hours of VoxForge dataset. Such a way, we are solving the issue, where low-resourced languages like Kazakh due to the inexistence of speech data cannot be researched and investigated in speech recognition area.

6.1 Methodology

The main objectives for development of ASR system for Kazakh language is to construct the speech corpus and to build a neural network with all advanced technologies. The well performance of a trained model is directly dependent on several factors:

- Training technique, after constructing the neural network
- Fine-tuning of neural network for optimization purposes
- An existence of a big speech corpus on defined language

There are many training techniques based on recurrent neural networks that we have discussed in above sections. They vary based on the given speech recognition task and all of them considered to be advanced technologies. They are Connectionist Temporal Classifier based RNN systems, RNN-transducer based RNN and Attention based RNN systems. These techniques are different from each other mostly based on their training methodology. CTC based systems doesn't require much compared to RNN-transducer systems and Attention based systems, since there is no need for building a language model for linguistic knowledge. It makes a dependency assumption between input sequence and output sequence, making a soft alignment between them. Despite the fact that, Attention-based systems are considered to be mostly used in speech recognition research community, CTC based ASR system for Kazakh language is more preferable, since an attention system requires tremendous amount of speech data. Kazakh language is considered to be as low-resourced language and therefore the usage of advanced technologies like Attention based techniques are not reasonable.

For building a proper ASR system for Kazakh language, the most important objective out of all is construction of speech corpus. As we have explored and discussed in above sections, building a speech corpus is not an easy task, especially for unpopular language like Kazakh. However, there are several methods on collecting speech data for a particular language and most of them are not suitable in terms of quality and quantity. In the sections above, we explained our new crowd-sourcing instrument for any languages that are suffering from lack of decent speech corpus. Using our data collection tool we have partially solved this problem. Moreover, we have applied another technique to that solves the speech corpus deficit entirely and it is transfer learning. We have taken the model which was trained on Russian language and applied to our system as a starting point. Since we are using CTC based model, the usage of completely different language like Russian is very reasonable since they share 78% of an alphabet. The following section explores and discusses the transfer learning topic broader and our approach on applying this method.

6.1.1 Transfer learning on speech recognition

Transfer learning is a novel approach that makes the training a lot better and accurate by transferring the knowledge from different task to a current task (Figure 6.1). Just as human beings, any model can be trained, learned significantly faster and efficient, if it had a previous experience on related tasks. Usually, source model is the model that has been trained on a massive amount of data, whereas the target model can be trained on small amount. Therefore, transfer learning can solve a problem with a lack of data. Nowadays, there are a lot of pre-trained models that are open-sourced and can be accessed very easily. The main idea behind transfer learning is to transfer the features and parameters (weights) from source task to target task. Basically, source model is considered as a starting point for a target model.

There are a lot of researches being done on multilingual speech recognition tasks [116, 117, 118].

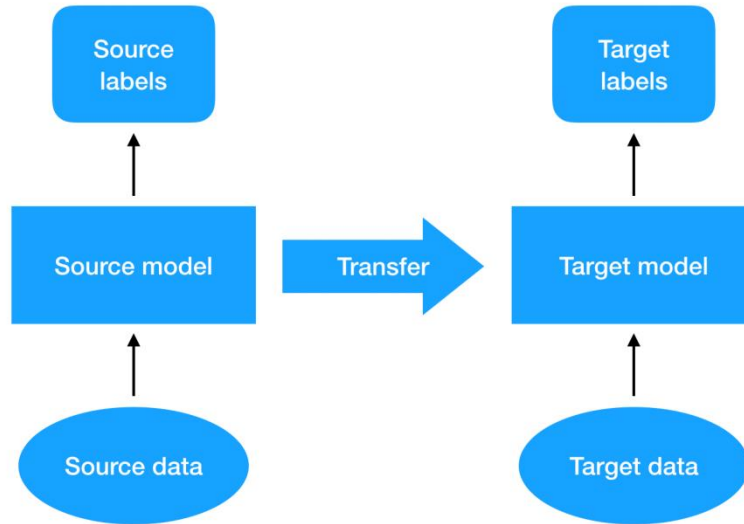


Figure 6.1 - High-level representation of transfer learning

The most common problem that they are facing till these days is that languages are specific. Language adaptive acoustic models can be built if languages share acoustic, pronunciation, phonetic properties.

This [119] paper presents the interesting approach on different region specific Indian 9 Languages, where each of these languages share the same phonetic, acoustic features and properties. They have combined all grapheme sets and trained on a sequence-to-sequence model. The result they have gained was 21% improvement on performance compared to the same model trained individually for each language.

For English, German and Spanish data from Cortana there has been an experiment presented here [120]. They have built a multilingual ASR system by using the universal character set shared around all languages. Each language had 150 hours of training set and 10 hours of validation and test set. By using [121], they have resulted 81 English labels, 93 German label and 97 Spanish labels. Therefore, they have built a universal label set (108) with 75% shared overlapping labels. By creating the mechanism which is called the language specific gating mechanism they have trained their model, which outperforms the monolingual approach.

In this [122] paper they have applied the transfer learning to a Text-to-Speech task, which is able to generate an audio with data, that has never seen before. They have uses 3 different pre-trained components: 1) *speaker encoder network* that has been trained on noisy speech dataset with no transcriptions; 2) *synthesis network* which is trained to generate mel spectrogram to text; 3) *vocoder network* which converts mel spectrogram to the waveform with time domain.

This [123] paper applies the transfer learning approach on developing ASR system for German language. With a limited training data, they have adapted a Wav2Letter

model, which is originally trained on English language. The paper [10] presented by Vu and Schultz have developed multilingual Multilayer Perceptrons (MLP). This MLP later was applied as a starting point on target languages like Vietnamese, Czech and Hausa. As a result, they have obtained the improvement on error rate up to 22%.

The approach that we take is almost the same as [123, 124], in which we are developing an ASR system for Kazakh language with the help of Russian language. The pre-trained model for Russian language has been trained on VoxForge dataset having a neural network structure with 2 LSTM layers with 128 neurons each and dense layer. Neural network has been trained with 700 epochs and has the same model with Bidirectional LSTM. It uses the same CTC loss function (Figure 6.2).

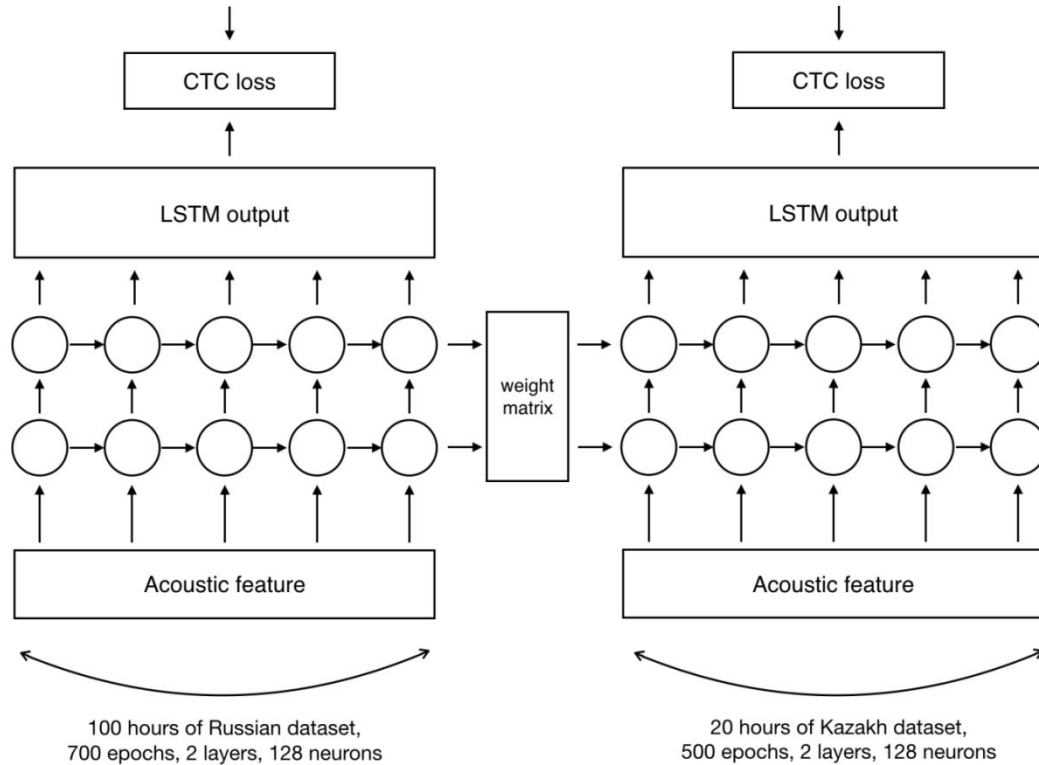


Figure 6.2 - High-level representation of weight transferring

Basically, we are using a pre-trained model to extract the weight matrix and copy to our exactly the same neural network structure with 2 LSTM layers. After application of “fine-tuning” our neural network, we start to train 20 hour of Kazakh speech dataset. Simply, we are initializing the weights of our neural network.

For fine-tuning the neural network, we have applied the optimization techniques like dropout for each layers, we have increase the batch size in order to make the training process faster, we have applied other optimizers like Momentum optimizer. Moreover, we have increased the number of layers and changed the type of RNN for comparison purposes and we also have been varying the number of epochs to see the influence.

6.2 Datasets

The speech data for this ASR system was collected in the base of Suleyman Demirel University. Overall dataset consisted 50 hours of speech. 20 hours of this corpus was collected manually from news portals and Kazakh books. The sentences from these sources were manually scrubbed. These transcriptions were provided to 50 native speakers that are around 23-25 years of age. The speech pronunciations of these transcriptions were collected without any platform or tool. It was purely based on the methods and environments of the individual speaker. It had many drawbacks since the condition of recordings process wasn't controller. Moreover, there weren't any clear instructions related to data collection provided. Therefore, many audio clips of this process were corrupted and had to be removed from overall corpus. After gathering the whole audio clips from speakers, all audio files were structured such a way, that they were corresponded with their transcriptions. In addition, audio clips with duration less than 2 seconds and over 15 seconds were omitted from corpus, since it greatly influences the strong dependency between audio signal and transcription. All audio files have been sampled to one frequency of 16kHz. Text files with transcriptions were normalized and processed in order to remove all unnecessary symbols that have no effect on recognition process.

Other 30 hours of speech data were collected using our web-based instrument called "Kazakh-recorder". As we have discussed in above sections, this is a web site that provides well designed platform to properly record the presented sentences. The sentences were provided from famous Kazakh literature books. The web application has a script that automatically processes the uploaded text file in any format and splits the sentences, removing unnecessary symbols and discarding too short and too long sentences. In data collection process, 65 users were involved that were native Kazakh speakers. Most of them are students around 18-23 years of age. Overall, 8 books were used to extract the sentences from, 20441 audio recordings were collected from users, 2030 audio recordings were corrupted and removed during the collection process, 18411 clear audio recordings were obtained after manually checking each and every clip, the duration of an entire process lasted approximately 1.5 month, entire audio clips were sampled into 16 kHz frequency, automatically around 500 recordings were neglected by an application due to the durations of audio clip. Manually collecting audio speeches previously was lasted almost 6 months. In contrast, our developed web application automates almost all process of data collection, which leads to better quality and increase of the speech data size.

Overall, having 100 hours of Kazakh speech data with corresponding transcriptions is almost enough to conduct proper experiment. The speech corpus is properly constructed, stored in GitHub repository and split into train set, validation set and test set (80%, 10%, and 10%).

6.3 Technologies

To conduct the experiment we have used Jupiter notebook environment, since it is very popular. The main language using which the entire code was developed is Python, due to its convenience and popularity around research community. Python language is easy to learn and compared to other programming languages like Java, C# and C++ considered being higher level.

In order to work with machine learning algorithms we have used an open-sourced Tensorflow library. It is frequently used for prediction task, training large amount of data and it is based on Python language, which makes the integration process very easy. It was invented at Google, which is known for its well-developed AI industry. Tensorflow gives the ability to developers to construct the data graphs, which described the data movement through nodes. Every node inside the graph is presented as math operation and the connection between these nodes are called tensors. Moreover, Tensorflow based applications can easily be processed in any type of machine like cloud services, local CPU's and GPU's. In addition, any model after the training process can be saved and applied in any platform (iOS, Android, web etc).

For building and constructing a neural network was utilized high level API of Tensorflow called Keras. The reason behind choosing this extension is the simplicity and convenience. It allows constructing any type neural network regardless of complexity. Moreover, it is based on Python language, which makes it to be easy to learn and friendly framework. Simple dense layer networks, more complex sequential networks and all types of network optimizers can be applied to any tasks. An entire neural network can be constructed using Keras library in few lines of code.

6.4 Experiment

Since the speech recognition system is considered to be sequence-to-sequence, to construct our neural network we are using RNN based network. We have conducted to our experiment 2 different RNN types: Long-Short Term Memory (LSTM) and bidirectional LSTM. Since BiLSTM in recent researches showed promising results and understands context very well, we have decided to compare it with regular LSTM. As we have mentioned above, as an environment for this experiment we have utilized Jupiter Notebook on Python programming language. Specifically, we have used Python based Tensorflow library to create and train the model for ASR. Moreover, to construct the neural network itself Keras library extension of Tensorflow has been used. The speech dataset is cloned from GitHub repository. The repository consists of three folders: train-set, validation-set and test-set. The training was done on several Graphical Processors (GPU) Tesla K80. The server with these processors has been rented for three month of usage. The list of parameters that were used in our experiment is the following (Table 6.1):

Table 6.1 - Parameters of neural network

Layers	2 layers of LSTM and BiLSTM separately
Units	128 neurons
Epochs	500
Optimizations	Dropout layer with 50%
Batch size	4
Optimizer	Momentum optimizer is 0.9
Learning rate	0.0005
Loss function	CTC loss
Metric	Label Error Rate (LER)

The pre-processing step is done for an audio file and text file. The reading process of an audio file is handled using librosa library. We have chosen “wav” format for audio clips, since the reading process (Appendix A) is easier than other format like mp4, mp3. As an audio feature extraction technique we have used Mel-frequency cepstral coefficients (MFCC), since it has valuable information about the audio and considered to be as one the most popular feature extraction methods in signal processing area. To extract MFCC we have utilized “python_speech_features” library, since it is easy to use. Every audio input is standardized before going into neural network (Appendix B). In order to avoid value error we have transformed the input sequence into the same length (Appendix C). As a result, it returns the array of element with the same dimensions. To pre-process the text we have used a simple python scripts with their internal models. To properly read all text files and work with folder we have utilized the “glob” library (Appendix D). To encode the text to numbers, we have constructed the dictionary of a kazakh alphabet with an extra “space symbol” (Appendix E). The decoding process of text is done using the same dictionary (Appendix F).

To restore the weights of Russian model, which was trained on 100 hours of Russian speech for 700 epochs we used the lines of code presented in Appendix G. It simply places all weight to the corresponding layers, because the structure of neural network that was made for Russian language is identical with our structure of network.

6.5 Results and evaluation

We have considered 4 different scenarios: 1) LSTM neural network without using Russian language model; 2) LSTM neural network using Russian language model; 3) Bidirectional LSTM without using Russian language model; 4) Bidirectional LSTM using Russian language model. They have used the same amount of layers and the same amount of neurons in each layer. The outcomes of the training process can be evaluated in Table

6.2. The result of each recurrent neural network actually very close, but we see that the architectures with transfer learning clearly make an improvements on everything.

LSTM layered neural network with external model has improved the training cost up to 8%, whereas Label error rate has increased up 4%. Bidirectional LSTM has showed very promising results, improving the training cost up to 24% and decreasing the label error rate down to 32% using the audio feature MFCC (Figure 6.5).

The experiments above showed that using an external Russian ASR system model to transfer its knowledge to Kazakh language system improves the performance decently. Considering the fact that the quality of VoxForge dataset is not decent enough, we understand that the overall performance can be increased even higher.

Table 6.2. Results of training

RNN type	Training cost	Training LER	Validation cost	Validation LER	Epochs
LSTM	3.987	0.05468	15.60	0.0156	500
LSTM with Russian model	4.789	0.05631	14.42	0.0145	500
BiLSTM	4.259	0.06285	18.36	0.0160	500
BiLSTM with Russian model	3.875	0.04272	13.92	0.01417	500

The visualization of learning process can be observed in Figure 6.3 and 6.4. Compared to other optimizers that we have used for this experiment like Adam and Adagrad, Momentum optimizer showed the best performance constantly decreasing the CTC loss and label error rate. Overall maximum duration of training had lasted more than 72 hours. We have reduced the training duration by applying fine-tuning to the network, slightly increasing the batch size and decreasing the learning rate.

For a CTC decoding part we have used both greedy search and beam search. As a result we didn't notice any differences. However, exploring many papers and works we have decided to use beam search algorithm (Appendix G).

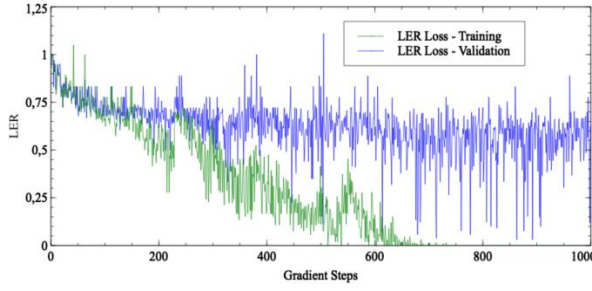


Figure 6.3 - LER illustration

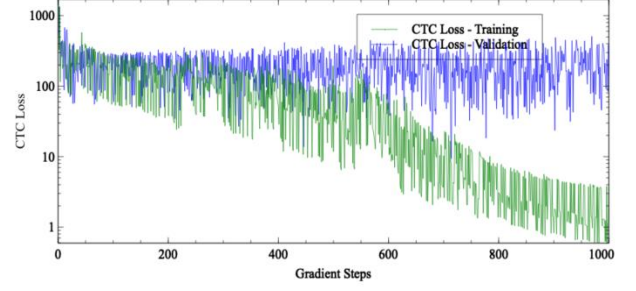


Figure 6.4 - CTC loss illustration

6.6 Discussion

The languages that are unpopular around the world and considered to be low-resourced, usually suffer from lack of investigation in speech recognition area. In order to build a proper ASR system for any language, construction of a neural network and fine-tuning is secondary objective. The most important objective is decent speech corpus that will allow the system to perform more effectively and powerfully. Since Kazakh language is recognized as unpopular language and collected speech corpus compared to other popular languages is considered to be very small, some extra manipulation with datasets must be done.

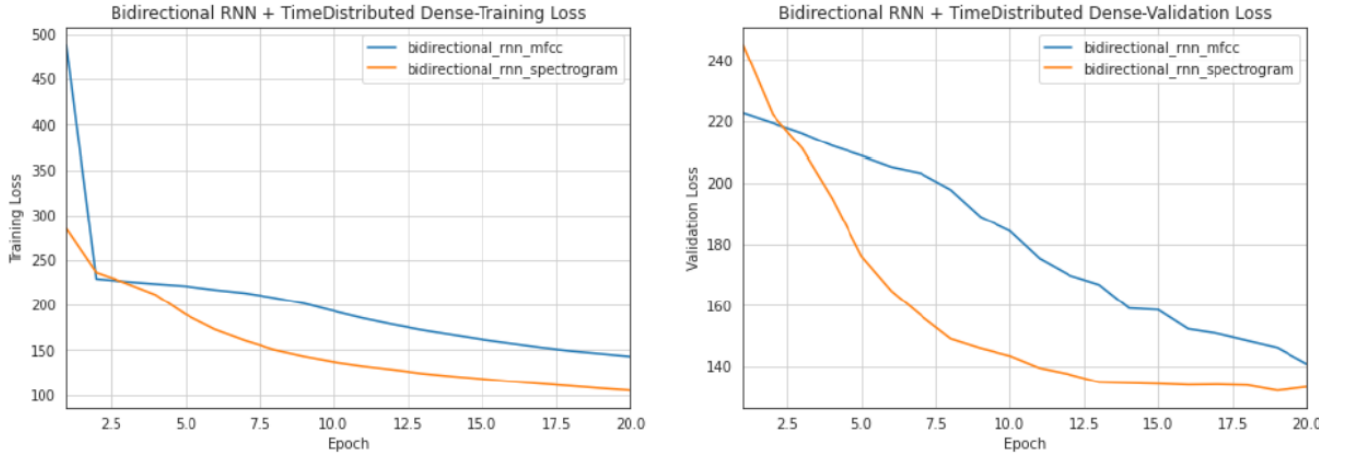


Figure 6.5 – The performance of BiLSTM

The transfer learning techniques are successfully applied to ASR system based on different languages; we have decided to apply the same approach to our native Kazakh language. It was initially utilized in image recognition area, in order to transfer the knowledge of the one model that was trained on one dataset to another model that is going to be trained on completely different dataset. The main idea behind this method is that there is no need for tremendous amount of data to start from scratch. As we have explored

in above sections, the knowledge transferring was applied in various different languages like German, Spanish, and Indian etc. They all have been successfully avoiding the massiveness of speech data, to gain the efficiency of the recognition model.

In our system, we utilized the Russian model, in order to copy its weights to our system. Although, these two languages (Kazakh, Russian) considered being completely different in terms of linguistics, they use very similar alphabet, that has in common over 75% of symbols. This transfer learning approach can produce a lot of benefit in many pattern recognition areas, especially in speech recognition field.

CONCLUSION

In the process of researching the speech recognition area all initiated objectives and tasks were obtained:

- The platform with well-designed environment and monitoring system for collecting speech data was constructed.
- Using the speech data collecting instrument over 100 hours of dataset was collected
- Transfer learning approach was applied to Kazakh ASR system using Russian speech recognition pre-built model

Using the speech collecting environment 65 native speakers were involved and over 50 hours of clean speech data was obtained in less than 1.5 month.

Kazakh speech recognition system was trained using neural network based on LSTM, BiLSTM layers. The application of multilingual approach using transfer learning (Russian pre-built model) has improved the performance of Kazakh ASR model by 24% in terms of Label Error Rate.

In conclusion, the problem of speech corpus has been solved using new approach of data collection. It allowed building our speech corpus in short amount of time using decent amount of native speakers. Also, the speech recognition model performed better using novel approach of transferring the knowledge from different language.

REFERENCES

- 1 S. Dixit and Y. Mulge, “Review on speech enhancement techniques”, *International Journal of Computer Science and Mobile Computing*, vol. 3, pp. 285–290, 2014.
- 2 G. Prasad, “A review of different approaches of spectral subtraction algorithms for speech enhancement”, *Current Research in Engineering, Science and Technology (CREST) Journals*, vol. 1, pp. 57–64, 2013.
- 3 A. Chaudhari and S. Dhonde, “A review on speech enhancement techniques”, *International Conference on Pervasive Computing (ICPC)*, 2015.
- 4 P. Bactor and A. Garg, “Different techniques for the enhancement of the intelligibility of a speech signal”, *International Journal of Engineering Research and Development*, vol. 2, pp. 57–64, 2012.
- 5 Y. Ephraim, H. Lev-Ari, and W. J. Roberts, “A brief survey of speech enhancement”, *IEEE Sig. Proc. Let.*, vol. 10, pp. 104–106, 2003.
- 6 L.-y. SUI, X.-w. ZHANG, J.-j. HUANG, and B. ZHOU, “An improved spectral subtraction speech enhancement algorithm under non-stationary noise”, *Institute of Command Automation, PLAUST Nanjing, China, IEEE*, 2011.
- 7 R. D.R, “Speech recognition by machine: A review”, *Proceedings of IEEE*, vol. 64, 1976.
- 8 J. W. PICONE, “Signal modeling techniques in speech recognition”, *Proceedings of IEEE*, vol. 64, 1993.
- 9 H. Georg, I. Moreno, S. Bengio, and N. Shazeer, “End-to-end text-dependent speaker verification.”, *Acoustics, Speech and Signal Processing (ICASSP)*, 2016.
- 10 N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, “Frontend factor analysis for speaker verification”, *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, 2011.
- 11 O. Thyges, R. Kuhn, P. Nguyen, and J. C. Junqua, “Speaker identification and verification using eigenvoices.”, *Sixth International Conference on Spoken Language Processing*, vol. 64, 2000.
- 12 A. Dhommne, R. Kumar, and V. Bhan, “Gender recognition through face using deep learning”, *Procedia Computer Science*, vol. 132, pp. 2–10, 2018.
- 13 H. H. Liu, S. S. D. Xu, C. C. Chiu, and S. Y. Chiu, “Gender recognition technology of whole body image”, *International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, 2017.
- 14 A. Kumar, K. Thankachan, and M. M. Dominic, “Sign language recognition”, *International Conference on Recent Advances in Information Technology (RAIT)*, 2016.
- 15 T. G. Hudson and S. Jaf, “On the development of a large scale corpus for native language identification”, *Proceedings of the 17th International Workshop on Treebanks and Linguistic Theories*, pp. 115–207, 2018.

- 16 S. Sunny, D. P. S., and K. P. Jacob, "Feature extraction methods based on linear predictive coding and wavelet packet decomposition for recognizing spoken words in malayalam", International Conference on Advances in Computing and Communications, 2012.
- 17 N. S. Nehe and R. S. Holambe, "Dwt and lpc based feature extraction methods for isolated word recognition", EURASIP Journal on Audio Speech and Music Processing, 2012.
- 18 G. B. Janvale, R. R. Deshmukh, and S. Gambhire, "Speech feature extraction using mel-frequency cepstral coefficient (mfcc)", the Emerging Trends in Computer Science, Communication and Information Technology, 2010.
- 19 S. Gupta, J. Jaafar, W. F. W. Ahmad, and A. Bansal, "Feature extraction using mfcc", Signal Image Processing : An International Journal (SIPIJ), vol. 4, 2013.
- 20 L. Muda, M. Begam, and I. Elamvazuthi, "Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques", JOURNAL OF COMPUTING, vol. 2, 2010.
- 21 D. A. Reynolds, "Gaussian mixture models", Encyclopedia of Biometrics, 2009.
- 22 T. T. Cai, C.-H. Zhang, and H. H. Zhou, "Optimal rates of convergence for covariance matrix estimation", Annals of Statistics, vol. 38, 2010.
- 23 L. M'enard, B. L. Davis, L.-J. Boe, and J.-P. Roy, "Producing american english vowels during vocal tract growth: A perceptual categorization study of synthesized vowels", Journal of Speech Language and Hearing Research, 2009.
- 24 T. Moon, "The expectation-maximization algorithm", IEEE Signal Processing Magazine, 1996.
- 25 J. Tejedor, J. Macias-Guarasa, H. F. Martins, S. Martin-Lopez, and M. Gonzalez-Herraez, "A gaussian mixture model-hidden markov model (gmmhmm)-based fiber optic surveillance system for pipeline integrity threat detection", 26th International Conference on Optical Fiber Sensors, 2018.
- 26 J. Zupan, "Introduction to artificial neural network (ann) methods: What they are and how to use them", Acta Chimica Slovenica, 1994.
- 27 G. Sivaram and H. Hermansky, "Sparse multilayer perceptron for phoneme recognition", IEEE Transactions on Audio Speech and Language Processing, 2012.
- 28 A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks", IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, pp. 328–339, 1989.
- 29 T. Robinson and F. Fallside, "A recurrent error propagation network speech recognition system.", Computer Speech Language., pp. 259–274, 1991.
- 30 T. Robinson, "A real-time recurrent error propagation network word recognition system.", Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 617–620, 1992.

- 31 T. Robinson and F. Fallside, “Word recognition from the darpa resource management database with the cambridge recurrent error propagation network speech recognition system.”, 1990.
- 32 A. Robinson, “An application of recurrent nets to phone probability estimation.”, *IEEE Transactions on Neural Networks*, vol. 5(2), pp. 298–305, 1994.
- 33 A. Hagen and A. Morris, “Recent advances in the multi-stream hmm/ann hybrid approach to noise robust asr.”, *Computer Speech Language*, vol. 19(1), pp. 3–30, 2005.
- 34 T. Pavelka and P. Kral, “Neural network acoustic model with decision tree clustered triphones.”, In *Proceedings of IEEE Workshop on Machine Learning for Signal Processing*, pp. 216–220, 2008.
- 35 H. Sak, A. Senior, and F. Beaufays, “Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition.”, 2014.
- 36 N. B. Lewandowski, J. Droppo, and M. Y. D. Seltzer, “Phone sequence modeling with recurrent neural networks.”, *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014.
- 37 G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, and N. Jaitly, “Deep neural network for acoustic modeling in speech recognition.”, *IEEE Signal Processing Magazine*, pp. 82–97, 2012.
- 38 A. R. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks.”, *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20(1), pp. 1–10, 2012.
- 39 A. R. Mohamed, G. Hinton, and G. Penn, “Understanding how deep belief networks perform acoustic modelling.”, *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4273–4276, 2012.
- 40 G. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition.”, *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20(1), pp. 30–42, 2012.
- 41 S. M. Siniscalchi, D. Yu, L. Deng, and C. H. Lee, “Speech recognition using long-span temporal patterns in a deep network model.”, *IEEE Signal Processing Letters*, vol. 20(3), pp. 201–204, 2013.
- 42 J. Gehring, W. Lee, K. Kilgour, I. Lane, Y. Miao, and A. Waibel, “Modular combination of deep neural networks for acoustic modeling.”, *Proceedings of Interspeech*, 2013.
- 43 D. Palaz, R. Collobert, and M. M. Doss, “Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks.”, *Proceedings of Interspeech*, 2013.
- 44 A. Graves, A. R. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks.”, *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013.

- 45 M. Mimura, S. Sakai, and T. Kawahara, “Reverberant speech recognition combining deep neural networks and deep autoencoders.”, Proceedings of REVERB Workshop., 2014.
- 46 G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean, “Multilingual acoustic models using distributed deep neural networks.”, Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing., 2013.
- 47 S. Rico, H. Barry, and B. Alexandra, “Neural machine translation of rare words with subword units.”, In Proceedings of ACL., 2016.
- 48 K. Philipp, J. O. Franz, and M. Daniel, “Statistical phrase-based translation.”, In Proceedings of NAACL., 2003.
- 49 B. Carlo, F. Enrico, P. Andrea, M. Simonetta, and S. Claudia, “Automatic semantics extraction in law documents.”, The Tenth International Conference on Artificial Intelligence and Law, Proceedings of the Conference, 2005.
- 50 J. Goodman, “A bit of progress in language modeling.”, Computer Speech and Language., vol. 15(4), pp. 403–434, 2001.
- 51 R. Miiikkulainen and M. Dyer, “Natural language processing with modular pdp neural networks and distributed lexicon.”, Cognitive Science., vol. 15(3), pp. 343–399, 1991.
- 52 J. Schmidhuber, “Sequential neural neural text compression.”, IEEE Transactions on Neural Network., vol. 7(1), pp. 142–146, 1996.
- 53 T. Mikolov, M. Karafiat, L. Burget, J. H. Cernocky, and S. Khudanpur, “Recurrent neural network based language model.”, Interspeech., vol. 2, pp. 1045–1048, 2010.
- 54 T. Mikolov, S. Kombrink, J. H. Cernocky, and S. Khudanpur, “Extensions of recurrent neural network based language model.”, IEEE International Conference on Acoustics, Speech, and Signal Processing., pp. 5528–5531, 2011.
- 55 W. Jian and Z. Fang, “On enhancing katz-smoothing based back-off language model.”, Sixth International Conference on Spoken Language Processing., 2000.
- 56 W. Wayne and S. Issar, “A class based language model for speech recognition.”, Acoustics, Speech, and Signal Processing., 1996.
- 57 S. Hochreiter and J. Schmidhuber, “Long short-term memory.”, Neural Computation., pp. 1735–1780, 1997.
- 58 M. Sundermeyer, R. Schluter, and H. Ney, “Lstm neural networks for language modeling.”, Interspeech., pp. 194–197, 2012.
- 59 F. A. Gers and J. Schmidhuber, “Recurrent nets that time and count.”, Proceedings of the IEEE-INNS-ENNS International Joint Conference, vol. 3, pp. 189–194, 2000.
- 60 K. Cho, B. M. Van, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation.”, 2014.

- 61 T. Hori, S. Watanabe, Y. Zhang, and W. Chan, “Advances in joint ctcattention based end-to-end speech recognition with a deep cnn encoder and rnn-lm.”, 2017.
- 62 A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks.”, In Proceedings of the International Conference on Machine Learning, pp. 1764–1772, 2014.
- 63 S. Kim, T. Hori, and S. Watanabe, “Joint ctc-attention based end-to-end speech recognition using multi-task learning.”, In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing., pp. 4835–4839, 2017.
- 64 A. Graves, S. Fern´andez, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks.”, In Proceedings of the 23rd international conference on Machine learning., pp. 369–376, 2006.
- 65 F. Eyben, M. Wollmer, B. Schuller, and A. Graves, “From speech to letters using a novel neural network architecture for grapheme based asr.”, In Proceedings of the 2009 IEEE Workshop on Automatic Speech Recognition Understanding., pp. 376–380, 2009.
- 66 J. Li, H. Zhang, X. Cai, and B. Xu, “Towards end-to-end speech recognition for chinese mandarin using long short-term memory recurrent neural networks.”, In Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association., pp. 3615–3619, 2015.
- 67 A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, and A. Coates, “Deepspeech: Scaling up end-to-end speech recognition.”, 2014.
- 68 A. Hannun, A. Maas, D. Jurafsky, and A. Ng, “First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns.”, 2014.
- 69 H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition.”, 2015.
- 70 W. Song and J. Cai, “End-to-end deep neural network for automatic speech recognition.”, 2015.
- 71 Y. Zhang, M. Pezeshki, P. Brakel, S. Zhang, C. Laurent, Y. Bengio, and A. Courville, “Towards end-to-end speech recognition with deep convolutional neural networks.”, 2017.
- 72 D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, and G. Chen, “Deep speech 2: End-to-end speech recognition in english and mandarin.”, In Proceedings of the International Conference on Machine Learning., pp. 173–182, 2016.
- 73 H. Soltau, H. Liao, and H. Sak, “Neural speech recognizer: Acoustic-to-word lstm model for large vocabulary speech recognition.”, 2017.
- 74 G. Zweig, C. Yu, J. Droppo, and A. Stolcke, “Advances in all-neural speech recognition.”, In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., pp. 4805–4809, 2017.

- 75 K. Audhkhasi, B. Ramabhadran, G. Saon, M. Picheny, and D. Nahamoo, “Direct acoustics-to-word models for english conversational speech recognition.”, 2017.
- 76 K. Audhkhasi, B. Kingsbury, B. Ramabhadran, G. Saon, and M. Picheny, “Building competitive direct acoustics-to-word models for english conversational speech recognition.”, In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., pp. 4759–4763, 2018.
- 77 R. Prabhavalkar, K. Rao, T. Sainath, B. Li, L. Johnson, and N. Jaitly, “A comparison of sequence-to-sequence models for speech recognition.”, In Proceedings of the Interspeech., pp. 939–943, 2017.
- 78 A. Graves, “Sequence transduction with recurrent neural networks.”, In Proceedings of the Interspeech., 2012.
- 79 V. Zue, S. Seneff, and J. Glass, “Speech database development at mit: Timit and beyond.”, Speech Communication., vol. 9, pp. 351–356, 1990.
- 80 K. Rao, H. Sak, and R. Prabhavalkar, “Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer.”, In Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)., pp. 193–199, 2017.
- 81 H. Sak, M. Shannon, K. Rao, and F. Beaufays, “Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping.”, In Proceedings of the INTERSPEECH., pp. 1298–1302, 2017.
- 82 L. Dong, S. Zhou, W. Chen, and B. Xu, “Extending recurrent neural aligner for streaming end-to-end speech recognition in mandarin.”, pp. 816–820, 2018.
- 83 D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate.”, 2014.
- 84 J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attentionbased models for speech recognition.”, pp. 577–585, 2015.
- 85 D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, “Endto-end attention-based large vocabulary speech recognition.”, In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., 2016.
- 86 K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation.”, 2014.
- 87 W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition.”, In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., pp. 4960–4964, 2016.
- 88 L. Lu, X. Zhang, and S. Renais, “On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition.”, In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., pp. 5060–5064, 2016.

- 89 C. Weng, J. Cui, G. Wang, J. Wang, C. Yu, D. Su, and D. Yu, “Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition.”, In Proceedings of the Interspeech ISCA., pp. 761–765, 2018.
- 90 C. Chiu, T. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. Weiss, K. Rao, and E. Gonina, “State-of-the-art speech recognition with sequence-to-sequence models.”, In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., pp. 4774–4778, 2018.
- 91 Y. Zhang, W. Chan, and N. Jaitly, “Very deep convolutional networks for end-to-end speech recognition.”, In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)., pp. 4845–4849, 2017.
- 92 J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, “End-to-end continuous speech recognition using attention-based recurrent nn: First results.”, In Proceedings of the NIPS 2014 Workshop on Deep Learning., 2014.
- 93 W. Chan and I. Lane, “On online attention-based speech recognition and joint mandarin character-pinyin training.”, In Proceedings of the Interspeech., pp. 3404–3408, 2016.
- 94 J. Hou, S. Zhang, and L. Dai, “Gaussian prediction based attention for online end-to-end speech recognition.”, In Proceedings of the INTERSPEECH., pp. 3692–3696, 2017.
- 95 T. Hayashi, S. Watanabe, T. Toda, and K. Takeda, “Multi-head decoder for end-to-end speech recognition.”, 2018.
- 96 S. Bougrine, A. Chorana, A. Lakhdari, and H. Cherroun, “Toward a webbased speech corpus for algerian dialectal arabic varieties.”, In Proceedings of the Third Arabic Natural Language Processing Workshop., pp. 138–146, 2017.
- 97 A. Salama, H. Bouamor, B. Mohit, and K. Oflazer, “Youdacc: The youtube dialectal arabic commentary corpus.”, Ninth International Conference on Language Resources and Evaluation, 2014.
- 98 A. Larcher, K. Lee, B. Ma, and H. Li, “Text-dependent speaker verification: Classifiers, databases and rsr2015.”, Speech Communication., vol. 60, pp. 56–77, 2014.
- 99 Y. Muthusamy, E. Holliman, B. Wheatley, J. Picone, and J. Godfrey, “Voice across hispanic america: A telephone speech corpus of american spanish.”, In Acoustics, Speech, and Signal Processing., vol. 1, pp. 85–88, 1995.
- 100 J. J. Godfrey, E. C. Holliman, and J. McDaniel, “Switchboard: Telephone speech corpus for research and development.”, In Acoustics, Speech, and Signal Processing., vol. 1, pp. 517–520, 1992.
- 101 I. Lane, A. Waibel, M. Eck, and K. Rottmann, “Tools for collecting speech corpora via mechanical-turk.”, In Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk., pp. 184–187, 2010.

- 102 K. A. Lee, “The reddots platform for mobile crowd-sourcing of speech data.”, In Sixteenth Annual Conference of the International Speech Communication Association, 2015.
- 103 D. A. Van Leeuwen, F. Hinskens, B. Martinovic, A. van Hessen, S. A. Grondelaers, and R. Orr, “Sprekend nederland. a heterogeneous speech data collection.”, 2016.
- 104 O. Mamyrbayev, M. Turdalyuly, N. Mekebayev, K. Mukhsina, A. Keylan, B. BabaAli, G. Nabieva, A. Duisebayeva, and B. Akhmetov, “Continuous speech recognition of kazakh language”, ITM Web of Conferences 24, 2019.
- 105 T. Schultz, “Globalphone: A multilingual speech and text database developed at karlsruhe university.”, In Seventh International Conference on Spoken Language Processing., 2002.
- 106 A. Nakamura, S. Matsunaga, T. Shimizu, M. Tonomura, and Y. Sagisaka, “Japanese speech databases for robust speech recognition.”, In Spoken Language., vol. 4, pp. 2199–2202, 1996.
- 107 J. Spitz, “Collection and analysis of data from real users: Implications for speech recognition/understanding systems”, In Speech and Natural Language., 1991.
- 108 E. Barnard, M. Davel, and C. Van Heerden, “Asr corpus design for resource scarce languages.”, 2009.
- 109 F. De Wet, P. Louw, and T. Niesler, “The design, collection and annotation of speech databases in south africa”, Proc Pattern Recognit. Assoc. South Afr., pp. 1–5, 2006.
- 110 T. Nadungodage, V. Welgama, and R. Weerasinghe, “Developing a speech corpus for sinhala speech recognition”, Proc Pattern Recognit. Assoc. South Afr., 2013.
- 111 K. Ronald and E. Barnard, “Statistical translation with scarce resources: A south african case study.”, 2006.
- 112 G. Parent and M. Eskenazi, “Speaking to the crowd: Looking at past achievements in using crowdsourcing for speech and predicting future challenges.”, In Twelfth Annual Conference of the International Speech Communication Association., 2011.
- 113 I. McGraw, C. Lee, I. L. Hetherington, S. Seneff, and J. R. Glass, “Collecting voices from the cloud.”, In LREC, pp. 1576–1583, 2010.
- 114 N. J. De Vries, “A smartphone-based asr data collection tool for underresourced languages.”, Speech Commun., vol. 56, pp. 119–131, 2014.
- 115 B. Li, “Acoustic modeling for google home.”, INTERSPEECH, pp. 399–403, 2017.
- 116 G. Arnab, S. Pawel, and R. Steve, “Multilingual training of deep neural networks.”, In Acoustics, Speech, and Signal Processing., 2013.
- 117 L. Hui, D. Li, Y. Dong, G. Yi-fan, A. Alex, and L. Chin-Hui, “A study on multilingual acoustic modeling for large vocabulary asr.”, In Acoustics, Speech, and Signal Processing., 2009.

- 118 T. Samuel, G. Sriram, and H. Hynek, “Multilingual mlp features for lowresource lvcsr systems.”, In Acoustics, Speech, and Signal Processing., 2012.
- 119 T. Shubham, S. Tara, W. Ron, L. Bo, M. Pedro, W. Eugene, and R. Kanishka, “Multilingual speech recognition with a single end-to-end model.”, 2018.
- 120 K. Suyoun and S. Michael L., “Towards language-universal end-to-end speech recognition.”, 2017.
- 121 Z. Geoffrey, Y. Chengzhu, D. Jasha, and S. Andreas, “Advances in all neural speech recognition”, 2017.
- 122 J. Ye, Z. Yu, W. Ron J., W. Quan, S. Jonathan, R. Fei, C. Zhifeng, N. Patrick, P. Ruoming, M. Ignacio Lopez, and W. Yonghui, “Transfer learning from speaker verification to multispeaker text-to-speech synthesis”, 2019.
- 123 K. Julius, K. Louis, K. Ilia, K. Andreas, J. Jens, and S. Sebastian, “Transfer learning for speech recognition on a budget”, Proceedings of the 2nd Workshop on Representation Learning for NLP, pp. 168–177, 2019.
- 124 V. Ngoc Thang and S. Tanja, “Multilingual multilayer perceptron for rapid language adaptation between and across language families”, Interspeech, pp. 515–519, 2013.

APPENDIX A – Reading audio files

```
defread_audio_files(dir, extensions=['wav']):

    if not os.path.isdir(dir):
        logging.error("Audio files directory %s is not found.", dir)
        return None

    if not all(isinstance(extension, str) for extension in extensions):
        logging.error("Variable 'extensions' is not a list of strings.")
        return None

    # Get files list.
    files_paths_list = []
    for extension in extensions:
        file_glob = os.path.join(dir, '*' + extension)
        files_paths_list.extend(glob.glob(file_glob))

    # Read files.
    files = []
    for file_path in files_paths_list:
        print(file_path)
        # audio_rate, audio_data = wav.read(file_path)
        # file = mfcc(audio_data, samplerate=audio_rate)
        # files.append(file)
        #-----

        audio_data, audio_rate = librosa.load(file_path)
        file = mfcc(audio_data, samplerate=audio_rate, nfft=1024)
        files.append(file)

    # audio_data, = librosa.load(file_path)
    # print(str(audio_rate)+"!!!!!!!!!!!!!!!!")
    # file = mfcc(audio_data, samplerate=8000, nfft=1024, lowfreq=50)
    # files.append(file)

    files = np.array(files)
    return files
```

APPENDIX B – Standardizing audio files

```
defstandardize_audios(inputs):
    """
    Standardize audio inputs.

    Args:
        inputs: array of audios.
        Audio files.
    Returns:
        decoded_text: array of audios.
    """
    result = []
    for i in range(inputs.shape[0]):
        item = np.array((inputs[i] - np.mean(inputs[i])) / np.std(inputs[i]))
        result.append(item)

    returnnp.array(result)
```

APPENDIX C - Transformation of sequence

```
def make_sequences_same_length(sequences, sequences_lengths, default_value=0.0):

    # Get number of sequences.
    num_samples = len(sequences)
    print(sequences)
    max_length = np.max(sequences_lengths)

    # Get shape of the first non-zero length sequence.
    sample_shape = tuple()
    for s in sequences:
        if len(s) > 0:
            sample_shape = np.asarray(s).shape[1:]
            break

    # Create same sizes array
    result = (np.ones((num_samples, max_length) + sample_shape) * default_value)

    # Put sequences into new array.
    for idx, sequence in enumerate(sequences):
        result[idx, :len(sequence)] = sequence

    return result
```

APPENDIX D – Reading all text files

```
defread_text_files(dir, extensions=['txt']):

    if not os.path.isdir(dir):
        logging.error("Text files directory %s is not found.", dir)
        return None

    if not all(isinstance(extension, str) for extension in extensions):
        logging.error("Variable 'extensions' is not a list of strings.")
        return None

    # Get files list.
    files_paths_list = []
    for extension in extensions:
        file_glob = os.path.join(dir, '*' + extension)
        files_paths_list.extend(glob.glob(file_glob))

    # Read files.
    files = []
    for file_path in files_paths_list:
        file = read_text_file(file_path)
        file = normalize_text(file)
        files.append(file)

    files = np.array(files)
    # print(files)
    return files
```

APPENDIX E – Dictionary of the alphabet

```
alphabet = {'a':1,  
            'ə':2,  
            'б':3,  
            'в':4,  
            'г':5,  
            'г':6,  
            'д':7,  
            'е':8,  
            'ё':9,  
            'ж':10,  
            'з':11,  
            'и':12,  
            'й':13,  
            'к':14,  
            'к':15,  
            'л':16,  
            'м':17,  
            'н':18,  
            'н':19,  
            'о':20,  
            'ө':21,  
            'п':22,  
            'р':23,  
            'с':24,  
            'т':25,'у':26,  
            'ү':27,'ү':28,'ф':29,  
            'х':30,'х':31,'ц':32,'ч':33,'ш':34,'ш':35,'ъ':36,'ы':37,'ї':38,'ь':39,'э':40,'ю':41,'я':42,  
            }
```

APPENDIX F – Decoding the text

```
defsequence_decoder(sequence, first_index=(alphabet['a'] - 1)):

    decoded_text = ".join([chr(x) for x in np.asarray(sequence) + first_index])
    # Replacing blank label to none.
    decoded_text = decoded_text.replace(chr(alphabet['я'] + 1), "")
    # Replacing space label to space.
    decoded_text = decoded_text.replace(chr(alphabet['a'] - 1), ' ')
    return decoded_text
```

APPENDIX G – Restoring the weights

```
checkpoint = tf.train.latest_checkpoint(checkpoint_dir=MODEL_PATH)
# Initialize the weights and biases.

current_epoch = 0
#print("!!!!!!!!!!!!!!"+checkpoint)
if checkpoint:
    print("[i] Loading checkpoint "+checkpoint)
    saver.restore(session, checkpoint)

    last_epoch = int(checkpoint.split('-')[-1]) + 1
    print("Start from epoch number-"+str(current_epoch))
else:
    tf.global_variables_initializer().run()
```


APPENDIX H – Dictionary of optimal characters

```
{  
  'a': [0,1],  
  'b': [2,3],  
  'c': [4,5]  
}
```

APPENDIX I – Extraction process

```
y, sr = librosa.load(fpath, sr=HyperParameters.sampling_rate)

# Trimming
y, _ = librosa.effects.trim(y)

# Preemphasis
y = np.append(y[0], y[1:] - HyperParameters.preemphasis * y[:-1])

# stft
linear = librosa.stft(
    y=y,
    n_fft=HyperParameters.n_fft,
    hop_length=HyperParameters.hop_length,
    win_length=HyperParameters.win_length
)

# magnitude spectrogram
mag = np.abs(linear) # (1+n_fft//2, T)

# mel spectrogram
mel_basis = librosa.filters.mel(HyperParameters.sampling_rate,
HyperParameters.n_fft, HyperParameters.n_mels) # (n_mels, 1+n_fft//2)
mel = np.dot(mel_basis, mag) # (n_mels, t)

# to decibel
mel = 20 * np.log10(np.maximum(1e-5, mel))
mag = 20 * np.log10(np.maximum(1e-5, mag))

# normalize
mel = np.clip((mel - HyperParameters.ref_db + HyperParameters.max_db) /
HyperParameters.max_db, 1e-8, 1)
mag = np.clip((mag - HyperParameters.ref_db + HyperParameters.max_db) /
HyperParameters.max_db, 1e-8, 1)

# Transpose
mel = mel.T.astype(np.float32) # (T, n_mels)
mag = mag.T.astype(np.float32) # (T, 1+n_fft//2)
```

APPENDIX J – Criffin-Lim algorithm

```
def griffin_lim(spectrogram):  
    "Applies Griffin-Lim's raw."  
    X_best = copy.deepcopy(spectrogram)  
    for i in range(HyperParameters.n_iter):  
        X_t = invert_spectrogram(X_best)  
        est = librosa.stft(X_t, HyperParameters.n_fft, HyperParameters.hop_length,  
win_length=HyperParameters.win_length)  
        phase = est / np.maximum(1e-8, np.abs(est))  
        X_best = spectrogram * phase  
        X_t = invert_spectrogram(X_best)  
        y = np.real(X_t)  
  
    return y
```