Dialect normalisation with deep learning-based automatic speech recognition

Author: Mahsa Vafaie

Supervisors: Inma Hernaez, Josef Van Genabith

Advisor: Jon Dehdari

Language Analysis and Processing

Master's Thesis

18th September 2017

Abstract

Automatic speech recognition has benefited a great deal from advances in machine learning throughout the past decade. Convolutional neural networks have demonstrated the ability to effectively model speech signals, exploiting temporal locality, but requiring very large amounts of input data. To date, these technological advances have been mainly applied to a small number of languages, such as English, German, Chinese and Spanish.

In this thesis, I implemented Speech-to-Text-WaveNet, a convolutional neural network architecture, on TFarsDat, a corpus of telephone conversations in Farsi, in order to test the replicability of deep learning-based techniques for an under-resourced language. While the final results of the system are far below state-of-the-art performance, the approach nonetheless demonstrates the viability of deep learning-based ASR, presupposing the existence of high-quality training data. I suggest that development of large, gold-standard datasets for under-resourced languages such as Farsi could be coupled with the proposed approach to improve the stat of ASR systems more generally.

Declaration

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged.

_____

Mahsa Vafaie

Tübingen

18th September 2017

Contents

List of Figures

List of Tables

1. Introduction

With computers becoming increasingly inseparable from our daily lives, more and more, computers enter into the practice of human communication. In general, human interaction with a computer either targets the computer as a responsive or non-responsive interlocutor (e.g. Siri, dictation software) or uses the computer as an intermediary in an interaction with another person (e.g. Skype, instant messaging). Norms from intra-human interaction have in many cases carried over into our interaction with computers: most often, a good computational interactant speaks or writes as humanly as possible.

Speech is the primary mode of human communication, figuring into diverse communicative genres, ranging from trivial everyday conversations to world-scale international negotiations. Unlike writing, which can be digitised and processed fairly easily, automated processing of human speech has proven a long-standing challenge. The potential benefits of automated recognition of human speech, however, are many: computers able to accurately recognise speech would potentially be able to transcribe, to follow directions, and to converse.

The field of *Automatic Speech Recognition (ASR)* has played a critical role in the ever-advancing field of human-computer interaction since the 1950s (Davis et al., 1952; Forgie and Forgie, 1959; Olson and Belar, 1956), with major breakthroughs almost each decade. The use of *Artificial Neural Networks (ANNs)*, intended to simulate the human brain, is one of the latest

trends in speech recognition, achieving results that out-perform earlier approaches (Lippmann, 1988). Deep learning has increased the accuracy of speech recognisers such that they can now be used outside of carefully controlled environments such as laboratories, and within consumer electronics such as personal computers, tablets and phones.

1.1. Automatic speech recognition

Traditionally, ASR systems are composed of two major components: the front end and the decoder. Figure 1.1 shows the block diagram of a traditional ASR system.

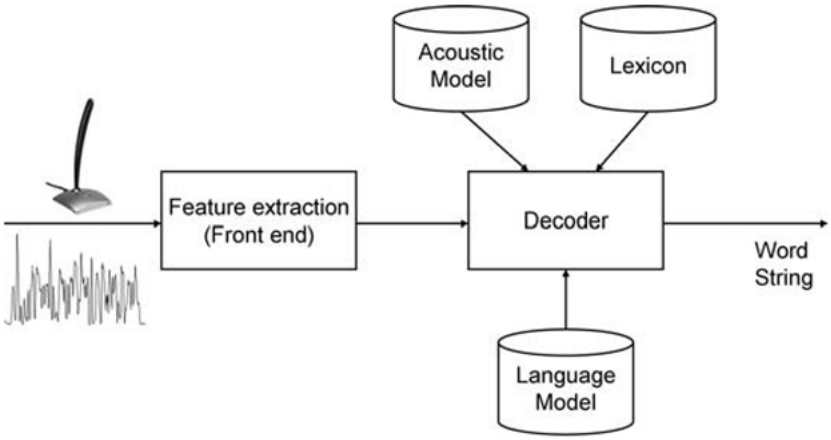

Figure 1.1.: A typical automatic speech recognition system

The front end builds a spectrum representation of the incoming speech wave. The most widely used features are *Mel Frequency Cepstral Coefficients (MFCCs)*, also used in neural speech recognition systems (see Section 2.2.1). The decoder block finds the best match of word sequences for the input acoustic features based on acoustic model, lexicon, and language model. The lexicon is a database that stores the pronunciations of words and their equivalent lexical representations/spellings, and the acoustic model pre-

dicts the sound units (i.e. phonemes) based on extracted speech features. The language model, in the final step, picks the best candidate among all, based on the context and neighbouring words.

Deep learning has made it possible to build end-to-end speech recognisers that take sound files, and directly turn them into transcriptions. These systems do not require all the components of a traditional ASR system, but instead, very large amounts of training data. This training data must contain a mapping of speech files to the expected outputs, which are usually either phonetically transcribed or standard-language written versions of the utterances. The general architecture of a deep learning-based speech recognition system is shown in Figure 1.2. It is a neural network fed with input, in this case audio files, trained to produce output in the form of text.

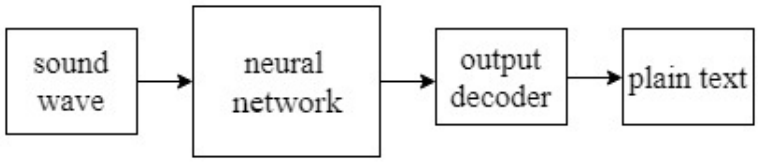

Figure 1.2.: A neural speech recognition system

Audio files contain recorded speech in the form of sound waves. Sound waves are one-dimensional entities with a single value at every moment, which is based on the height of the wave at that moment. Recording the height of the wave at equally-spaced points is called *sampling*. In a neural speech recognition system, the set of sound waves is transformed into a purely numerical representation, and then fed into the neural network. In general, a sampling rate of 16kHz (16,000 samples per second) is enough to cover the frequency range of human speech, and thus to suffice for speech recognition tasks.

To cope with the non-stationary nature of the speech signal, windows of 20 to 40 milliseconds are taken to extract the parameters. These parameters are a representation of the frequency components of the signal. Different representations have been reported in the literature, but MFCCs continue to be the most popular choice.

After the sound files are transmitted and fed into the neural network in chunks of 20- to 40 milliseconds, the neural network will try to predict which character(s) (phonemes or graphemes, depending on the task) correspond to each chunk. Using the parameters in the trained model, there will be a probability distribution over letters of the alphabet for each chunk. Given this distribution, the most likely sequence of characters is picked as the output sequence. This step, called *output decoding*, can be done using different algorithms. The resulting transcription then has to be cleaned before it is presented as output. For instance, take a case where the output sequence of characters for a piece of audio is `SSSA_LL_AAAAM`. First, any repeated string of characters is replaced with only one character, transforming `SSSA_LL_AAAAM` into `SA_L_AM`. Then the blanks are removed, and the result is presented as the final transcription, `SALAM` (*Hello*, in Farsi).

1.2. Motivation of the thesis

While speech recognition is constantly improving, and new tools and methods are under constant development, most work in the field to date has focused on a small number of languages, such as English, German, Chinese and Spanish. Out of more than 6,900 languages in the world, only a few are well-resourced enough for useful implementation of Human Language Technologies (HLT) (Besacier et al., 2014). Currently, ASR can be characterised as conforming or contributing to the "global digital divide" (see Chinn and Fairlie, 2007), where access to digital resources is concentrated

in a small set of wealthy states and linguistic groups. This unequal availability of technologies such as ASR can hinder technological growth and participation in the global economy for those without access, and function for those who control the technology as mechanisms of neocolonialism and control.

In the context of language and linguistics, as Besacier et al. (2014) suggest, the "language divide" can be improved by increasing the generalisability and portability of speech and language technologies for multilingual applications, especially for under-resourced languages. Part of the promise of neural network-driven ASR is that it is language-agnostic, requiring only training data in the target language in order to be successfully implemented in a novel language.

This thesis focuses on Farsi, a language that is widely spoken, but which is currently lacking in tools and resources for ASR. The work represents an attempt to both test the portability of an existing deep neural network ASR pipeline (*Speech-to-Text-WaveNet*—see Kim and Park, 2016), and to build and implement an open-source system for Farsi ASR.

1.3. The Farsi language

Farsi, also known as Persian, is the most widely spoken member of the Iranian branch of the Indo-Iranian languages within the Indo-European language family. Farsi is the official language of Iran, Afghanistan (where it is officially known as Dari), and Tajikistan (known as Tajiki, since the Soviet era) with approximately 110 million native speakers around the world, and with at least 20 recognised dialects (Gordon et al., 2005). In Iran, Farsi is written using the Persian alphabet, which is a modified version of Arabic alphabet. The Persian orthographic system consists of 32 graphemes for consonants and 6 graphemes for vowels. Some of these

graphemes represent one single phoneme in Farsi, but different phonemes in Arabic. Table A1 in the appendix, borrowed from Bijankhan et al. (2003), lists phonemes in Farsi, their corresponding IPA representation, orthographic representation in the Persian alphabet, and a brief phonetic gloss. Short vowels in Farsi (represented in IPA as *e*, *a* and *o*) are optional in written Farsi in unambiguous cases; for this reason, they are often elided. This results in a loose grapheme-to-phoneme mapping in the writing system, and thus poses a potential challenge to grapheme-based ASR.

Within Farsi dialects, there is also a great deal of phonological, lexical and grammatical variation. *Ketabi Farsi*, generally seen as "standard" or "formal" Farsi, is the main language of books and newspapers (*ketab*, in fact, means *book*). Colloquial Farsi (encompassing a number of regional dialects), on the other hand, is the common language of everyday interactions. The difference between these two modes is significant enough to cause serious challenges for NLP systems, trained on data from the standard written form. For instance, informal Farsi has shortened verbal stems and inflectional endings (Megerdoomian, 2006). Table 1.1 lists conjugations of the verb *goftan* (English: *to say*) in present tense, in formal and conversational Farsi (Tehrani accent). The first morpheme, *mi*, is a verbal prefix marking continuity. The second part, *gu*, is the stem, which is shortened and reduced to a single consonant, *g*, in conversational speech. The last part is the inflectional suffix marking person and number, also shortened in conversational Farsi.

State-of-the-art ASR systems for Farsi can recognise continuous speech in Ketabi Farsi (Sameti et al., 2011), but colloquial speech still poses a problem for these systems. Phonological/dialectal variations, for example, can result in *Out Of Vocabulary* words that are problematic for traditional speech recognition systems that rely on a lexicon.

Person	Number	Formal Farsi	Informal Farsi
1st	Singular	*mi-gu-yam*	*mi-g-am*
2nd	Singular	*mi-gu-yi*	*mi-g-i*
3rd	Singular	*mi-gu-yad*	*mi-g-e*
1st	Plural	*mi-gu-yim*	*mi-g-im*
2nd	Plural	*mi-gu-yid*	*mi-g-id*
3rd	Plural	*mi-ug-yand*	*mi-g-an*

Table 1.1.: Conjugation of *goftan* (*to say*) in formal and informal Farsi in Tehrani accent

1.4. Aim and scope of the thesis

The aim of this thesis is to build a deep learning-based ASR system that can recognise colloquial Farsi in different dialects and output a normalised phonetic representation in Ketabi Farsi. This is accomplished by repurposing two main resources:

1. **Speech-to-Text-WaveNet**, a speech-to-text pipeline that implements a convolutional neural network in TensorFlow
2. **TFarsDat**, a corpus of phonetically transcribed Farsi telephone conversations in both Ketabi and colloquial Farsi

A secondary aim is to evaluate the challenges encountered during this process, and to identify pathways toward higher quality speech-to-text systems for Farsi as an under-resourced languages.

The thesis is necessarily constrained in scope. First, despite the fact that dialects of Farsi vary at phonological, lexical and grammatical levels, normalisation of the data targeted only phonological variation. Second, the thesis focuses on using existing training data, rather than on development of a gold-standard collection of Farsi transcriptions built specifically for the purposes of dialect normalisation. Results are therefore bound to the quality of the existing dataset.

1.5. Research Questions

1. What are the major challenges in implementing a deep learning-based ASR pipeline for Farsi?

2. To what extent can an existing pipeline (namely, *Speech-to-Text-WaveNet*—see Kim and Park, 2016) and training dataset (*TFarsDat*—see Bijankhan et al., 2003), be repurposed for dialect normalisation in Farsi?

1.6. Overview of the thesis

Chapter 1 has introduced the context, aims, scope and research questions of the thesis, with special attention paid to Farsi, the target language of the developed ASR system.

Chapter 2 reviews past and present ASR research, framing state-of-the-art developments in a historical context. The chapter also provides an overview of available ASR research on Farsi.

Chapter 3 presents the datasets and tools used in this study, and describes the architecture of the developed speech recognition pipeline.

Chapter 4 presents the research design of three experiments, as well as their results and evaluation measures.

Chapter 5 outlines the contributions and limitations of the thesis, and discusses directions for possible future work.

2. Literature Review

Automatic computational recognition of human speech has long been of interest to researchers both in academia and industry. For this reason, the field has both a long history and a great deal of current activity. In this chapter, I first summarise key historical developments in the field of speech recognition. With this context established, I synthesise literature describing contemporary approaches to ASR and state-of-the-art techniques, and then present a short overview of existing ASR research for Farsi.

2.1. History of ASR

Research into ASR dates back to the early 1950s, when the first ASR system was made for single-speaker digit recognition by Bell Laboratories. This system worked by locating formant frequencies, which are manifested as major regions of energy concentration, in the power spectrum of each utterance, and matching them with patterns (Davis et al., 1952). Other early recognition systems from this period include the single-speaker syllable recogniser of Olson and Belar (1956) and Forgies' speaker-independent ten-vowel recogniser (Forgie and Forgie, 1959).

In the 1960s two researchers at Kyoto University employed a speech segmenter for the first time, making it possible to recognise and analyse individual words within an input utterance (Sakai and Doshita, 1962). As

the first project to focus on processing of natural language, rather than speech sounds or words in isolation, it may be seen as the first attempt at creating a continuous speech recognition system (Juang and Rabiner, 2005). Later in the decade, Soviet researchers devised the *Dynamic Time Warping (DTW)* algorithm and used it to create a recogniser capable of operating on a 200-word vocabulary (Velichko and Zagoruyko, 1970). The DTW algorithm, still in use today, divides the speech signal into short frames, e.g. 10 millisecond segments, and processes each frame as a single unit. Despite the breakthroughs associated with segmentation of speech signals, implementations in this period were constrained in scope, handling only specific phonemes or words; analysis of unconstrained continuous speech remained a distant goal.

During the early 1970s, the Defense Advanced Research Projects Agency (DARPA) of the U.S. Department of Defense started funding the Speech Understanding Research (SUR) programme, working in cooperation with a number of research institutes and private companies in the United States. IBM was focused on creating a "voice-activated typewriter", which converted a spoken utterance into a sequence of letters that could be typed on paper or displayed on a screen (Jelinek et al., 1975)—a task generally referred to as *transcription*. They built a speaker-dependent speech recognition system, equipped with a language model and a large vocabulary. At AT&T Bell Laboratories, on the other hand, the focus was on speaker-independent speech recognition. Their goal was to provide automated telecommunication services to the public. To achieve this, systems were needed that could successfully process input from speakers with different regional accents, without the need for individual speaker training. Their approach involved the development of speech clustering algorithms, which built word and sound reference patterns containing information about dialectal variants of a given phoneme. Research at Bell Laborator-

ies emphasised *keyword spotting* as a basic form of speech understanding (Wilpon et al., 1990). Keyword spotting is a technique aimed at detecting salient terms (i.e. ideationally rich content words) embedded in a longer utterance, while paying less attention to closed-class/function words (determiners, conjunctions, prepositions, etc.).

The introduction of *Linear Predictive Coding (LPC)* (Atal and Hanauer, 1971; Itakura, 1970) in the 1970s changed the approach toward input signals. LPC is a tool for representing the spectral envelope of a digital speech signal in compressed form, using the information of a linear predictive model (Deng and O'Shaughnessy, 2003). In the decades to come, extracting features from the speech signal before inputting them into the system became state-of-the-art technique, with feature extraction becoming a major research interest. Later in the 1980s, *Perceptual Linear Prediction (PLP)* coefficients (Hermansky, 1990) and *Mel-Frequency Cepstral Coefficients (MFCCs)* (Davis and Mermelstein, 1980) were introduced. These approaches are still in use today, including the experiments in this thesis. Section 2.2.1 below gives a detailed explanation of MFCCs.

The dominant approach to speech recognition in the 1970s was a template-based pattern recognition paradigm, in combination with acoustic-phonetic methods. This methodology shifted toward a statistical modelling framework in the 1980s. *Hidden Markov Models (HMMs)* became the preferred method for speech recognition and stayed so for long after the theory was initially published (Ferguson et al., 1980; Levinson et al., 1983). The use of HMMs allowed researchers to combine different sources of knowledge, such as acoustic models and language models, in a unified probabilistic model.

With the widespread use of HMM techniques, researchers realised that the performance of the system was being constrained by limitations on the form of the density functions. This was particularly harmful for speaker-

independent tasks. These limitations were partially overcome by extending the theory of HMM to mixture densities (Juang, 1985; Lee et al., 1990) to ensure satisfactory recognition accuracy, particularly for speaker-independent, large-vocabulary speech recognition tasks. This extension of the HMMs is called *Gaussian Mixture Models (GMMs)*.

In the late 1980s, *Artificial Neural networks (ANNs)* emerged as a new approach to acoustic modelling in ASR. Neural networks were first introduced by McCulloch and Pitts (1943) as an attempt to mimic the human neural processing mechanism. While the approach attracted little attention initially, it was revived in the 1980s with the advent of *Parallel Distributed Processing (PDP)*, known as *Connectionism* today. PDP is an artificial neural network approach that stresses the parallel nature of neural processing, and the distributed nature of neural representations. Lippmann (1988) reported success using neural networks for preliminary speech recognition in constrained tasks such as vowel classification and digit recognition. Later, Lippmann (1989) specifically described the potential for neural networks to offer new algorithmic approaches to problems in speech recognition.

In the 1990s Bourlard and Morgan (1993) built a hybrid speech recogniser, replacing the GMM with a single-layered neural network in an HMM-based system. This method successfully predicted the correct HMM, but the performance was still lower than the GMM-HMM architecture. The main reason for this were that large amounts of high-quality training data and computational resources capable of performing the large numbers of calculations necessary for neural networks were lacking at that time.

2.2. State of the art

The introduction of *Deep Neural Networks (DNNs)* in the past decade, coupled with growing affordability of high-performance computing, has resolved many historical challenges in ASR. A DNN is a feed-forward, artificial neural network that has multiple layers of hidden units between its inputs and its outputs. DNNs can approximate any function with desired accuracy given enough layers and nodes. Hinton et al. (2012) report the DNN approach showing significant improvements over Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) systems in a variety of state-of-the-art ASR systems.

Using DNNs to build complete ASR systems is one of the latest advances in ASR (Graves et al., 2006; Graves, 2012b,a; Graves and Jaitly, 2014). Graves and Jaitly (2014) combine a deep bidirectional LSTM network architecture with a *Connectionist Temporal Classification (CTC)* objective function (see Section 2.2.2) to build an end-to-end speech recogniser, which directly transcribes audio data to text without an intermediate phonetic representation. They report a word error rate of 27.3% on the Wall Street Journal corpus with no prior linguistic information, and 8.2% with a trigram language model.

2.2.1. Mel Frequency Cepstral Coefficients (MFCCs)

The first step in any machine learning task is to extract relevant features from data in order to reduce data complexity. Feature extraction in the context of ASR involves identifying the components of the audio signal that are characteristic of the linguistic content, while discarding all the other non-linguistic sounds that carry information like background noise, line noise, etc. The most commonly used feature extraction method in ASR

is *Mel-Frequency Cepstral Coefficients (MFCCs)*. MFCCs were introduced by Davis and Mermelstein (1980), and have been state-of-the-art ever since.

To extract features that contain all information about the linguistic content, MFCC mimics some parts of the human speech production and speech perception. Sounds generated by a human are filtered by the shape of the vocal tract. This shape determines what sound comes out. Subsequently, if we can determine the shape, it should give us a representation of the phoneme being produced. The shape of the vocal tract manifests itself in the envelope of the short time power spectrum, and MFCCs can accurately represent this envelope. They also eliminate speaker dependent characteristics by excluding the fundamental frequencies, which makes them suitable for speaker-independent speech recognition tasks.

The flowchart for implementation of Mel-Frequency Cepstral Coefficients algorithm is shown in Figure 2.1 below. The steps to compute MFCCs are as follows:

1. Frame the signal into short (10–25ms) frames.
2. For each frame calculate the periodogram estimate of the power spectrum.
3. Apply the mel filterbank to the power spectra, sum the energy in each filter.
4. Take the logarithm of all filterbank energies.
5. Take the Discrete Cosine Transform (DCT) of the log filterbank energies.
6. Choose the number of cepstral coefficients (typically in a range of 13 to 20) for further processing, keep them and discard the rest.

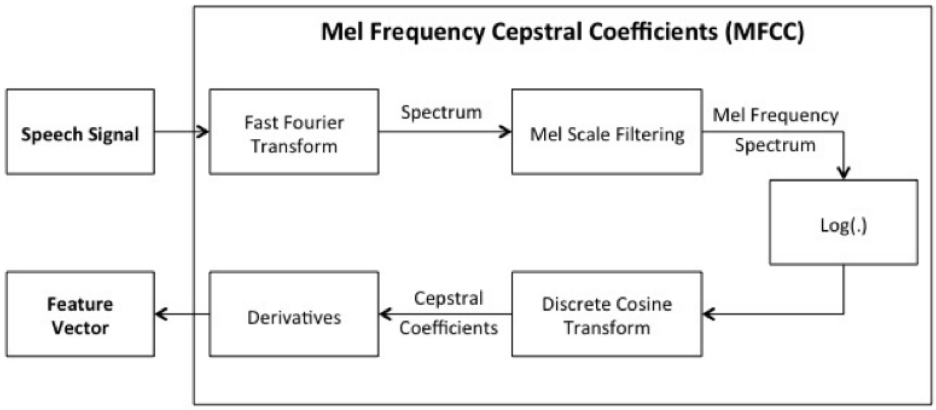

Figure 2.1.: Block diagram of the MFCC algorithm

2.2.2. Connectionist Temporal Classification (CTC)

As a sequence learning task, ASR requires a tool for prediction of labels from an unsegmented acoustic signal. Neural Networks need aligned input and output data in order to learn to classify and label the speech signals. *Connectionist temporal classification (CTC)*, first introduced by Graves et al. (2006), is the state-of-the-art sequence learning algorithm used with neural networks for labelling unsegmented data sequences. The approach involves transforming network outputs into a probability distribution over all possible label sequences, given an input sequence. An objective function is then defined that maximises the probability of the correct labelling, given the distribution. This optimisation problem can be solved using the forward–backward algorithm.

The CTC algorithm, explained in three steps below, removes the pre-segmentation and post-processing steps in an automatic speech recognition pipeline.

1. The ANN output neurons, called *c* here, encode a distribution over the symbols in the alphabet (see Figure 2.3). In a grapheme-based model,

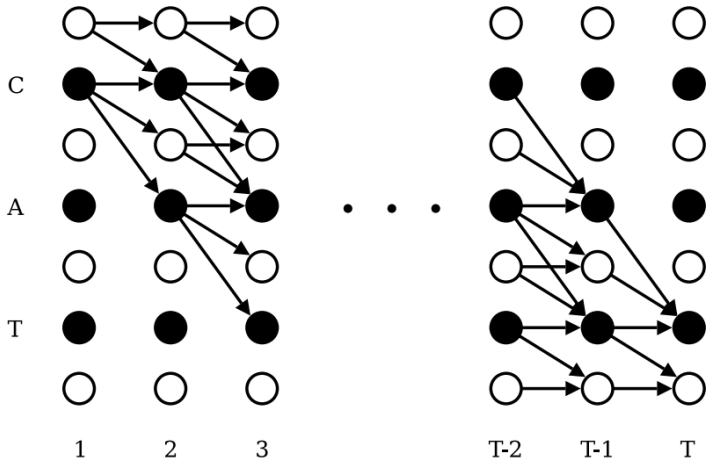

Figure 2.2.: Forward–backward algorithm applied to labelling of *CAT* (from Graves et al., 2006)

.

c takes on a value from the set union of the alphabet, blank, and space, if the language has spaces in it. In case of English this would mean $c \in \{A, B, C, ..., Z, blank, space\}$. With an independence assumption, the distribution of all possible sequences of characters over the alphabet can be defined by

$$P(c|x) \equiv \prod_{i=1}^{N} P(c_i|x) \tag{2.1}$$

For instance, the probability of the string `SSS-A-LL-AM-` is calculated by multiplying the probabilities of each of its characters given the input audio:

$$P(c|SSS--A-LL--AM--) \equiv P(c_1 = S|x)P(c_2 = S|x)...P(c_{15} = blank|x)$$

The symbol sequence *c* has the same length as the audio input *x*, but this might not be equal to the length of the transcription *y*.

2. The output sequence *c* is then mapped into the transcription *y* by a function β

$$\beta(c) \rightarrow y \tag{2.2}$$

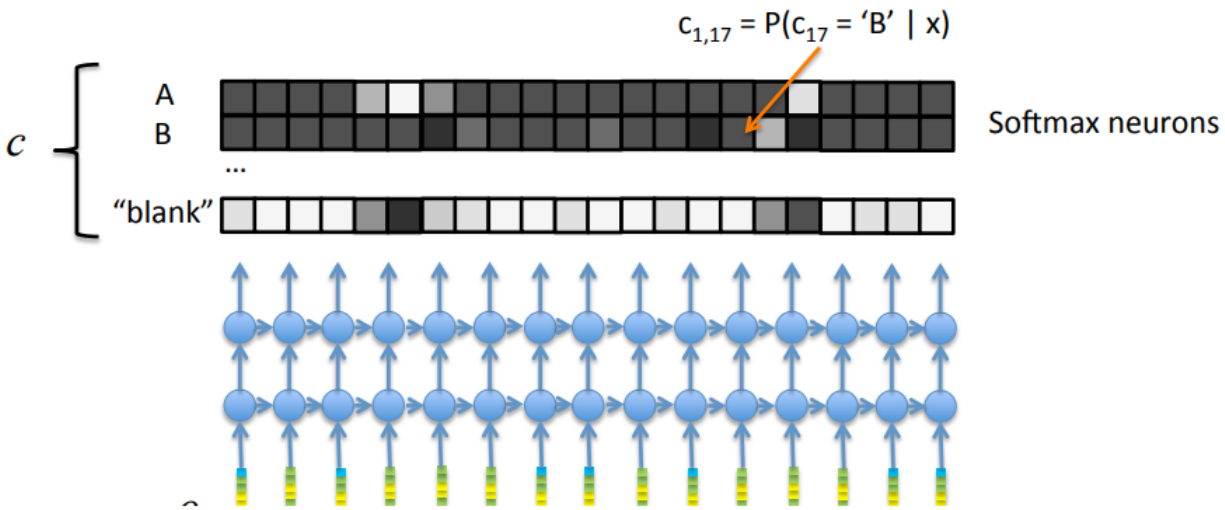

Figure 2.3.: Probability distribution of the output neurons c over the output alphabet given the spectogram representation of the input audio

This mapping function discards duplicates in a string of similar adjacent characters, and then drops all of the blanks in the character sequence c, to yield the transcription y.

$$y = \beta(c) = \beta(SSS--A-LL--AM--) = "SALAM"$$

Since c is a probabilistic entity, this mapping function will also result in a distribution over all possible final transcriptions y. Going through all possible combinations c, it might be the case that several of these combinations are mapped into a single transcription y. The probability of a transcription y is then computed by summing over all possible choices of c that correspond to y.

$$P(y|x) = \sum_{c:\beta(c)\to y} P(c|x) \tag{2.3}$$

This operation solves the problem of variable input length, caused by differences is speech rate, accent, or other factors.

3. Finally, network parameters θ are updated to maximise the likelihood of the correct transcription $y*$, given the audio input x, for all data points i.

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_i logP(y^{*(i)}|x^{(i)}) \tag{2.4}$$

Combining equations 3.3 and 3.4 will result in the following equation

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_i log \sum_{c:\beta(c)\rightarrow y^{*(i)}} P(c|x^{(i)}) \tag{2.5}$$

which says the probability of a given transcription is the summation of all possible symbol sequences that could have given that transcription.

Graves et al. (2006) provide a dynamic programming algorithm that can efficiently compute the summation and its gradient with respect to c, the output neurons of the neural network.

Given network outputs and target labels, there are software packages available that run the CTC algorithm and compute CTC loss from c, $y*$ and gradient with respect to c. CTC computes as cost function (or objective function), the negative log likelihood of the probability of the target transcription given the output of the neural network, while the output of the neural network is the probability distribution over the alphabet given the sound signal.

$$CTCloss = -logP(y^{*(i)}|c^{(i)}) \tag{2.6}$$

As the CTC loss decreases the probability that the parameters are set to fit the transcriptions increases. This makes CTC loss a good measure for tuning the hyperparameters and model selection.

2.2.3. Convolutional Neural Networks

Convolutional Neural Networks (CNNs) as an alternative type of neural networks, have shown benefits over other neural networks in the fields of computer vision, image recognition, and speech recognition, with their ability to model local correlations between input features (LeCun et al., 2004; Lawrence et al., 1997; Abdel-Hamid et al., 2012). A complete description of the different kinds of CNN is out of the scope of this document. However, in order to set the scene for implementation configurations, short descriptions are given together with some references which have described them in more details.

CNNs are a class of feed-forward deep neural networks that consist of one or more *convloutional* layers and optional *pooling* and *fully-connected* layers. Inspired by the mammalian visual cortex, CNNs use weights called *filters* (or *kernels*) that detect specific attributes. As the input progresses through each layer, the filters are able to recognise more complex attributes.

The fundamental layer in a convolutional neural network is a convolutional layer. The output of this layer is called a *feature map*. In order to generate a feature map, the filter which is an array/matrix of numbers slides through the input array/matrix and the element-wise production of the two arrays/matrices is placed in the corresponding slot in the feature map. This operation is called *convolution*. The number of data points between each convolution is called a *stride*.

Each convolution layer is conventionally followed by an *activation layer*, whose purpose is to introduce non-linearity to the system. The activation layer applies either tanh and sigmoid functions, or a function $f(x) = max(0, x)$, called a *Rectified Linear Unit(ReLU)*, to all of the values in the input volume.

The pooling layer in a CNN reduces the size of the feature map and simplifies computations in later layers by down-sampling the feature map. There are different functions for implementation of the pooling layer, including *max pooling*. Max pooling slides a window over the feature map, and at each slot, outputs the largest values in the window and discards the rest.

The fully-connected layer will eventually produce class scores from the output neurons which are used for classification. This layer performs the same duty in a CNN as in a standard ANN.

In a CNN each of these layers can be multiply stacked on one another. In addition to the order and number of the layers, there are other hyperparameters that can be modified in a CNN. The size and number of filters in the convolution layer, and the size of the window used in the max pooling layer, are some of these hyperparameters.

Recent uses of CNNs in language processing have yielded promising results. Sainath et al. (2013) applied CNNs to large vocabulary speech recognition tasks. They implemented a CNN with two convolutional and four fully-connected layers on 300 hours of conversational American English telephony data from the Switchboard Corpus, and reported 21.9% word error rate. Sainath et al. (2015) also proposeed various CNN architectures, with modifications in the number of convolutional and fully-connected layers, number of hidden units, filter sizes, type of pooling, and some other specifications, and reported a word error rate of 13.6% on 50 hours of English Broadcast News (BN) corpus, 12.7% on 400 hours of BN and 10.7% on 300 hours of Switchboard corpus.

Qian and Woodland (2016) developed a novel CNN architecture, termed Very Deep CNN (VDCNN) for noise robust speech recognition. They reported a word error rate of 13.52% on a subset of "noisy data with channel distortion" from *Aurora 4* task, without using front end denoising. Aurora

4 (Pearce and Picone, 2002) is a speech recognition task based on the Wall Street Journal Corpus (WSJ0). It consists of 16kHz speech data in the presence of additive noises and linear convolutional channel distortions, which were introduced synthetically to clean speech from WSJ0.

The optimal architecture of a CNN for large scale speech recognition has been subject of many other studies as well, mostly with English as the target language (Palaz et al., 2015; Song and Cai, 2015; Abdel-Hamid et al., 2014). This work borrows its architecture from WaveNet, a deep convolutional neural network originally designed for speech synthesis, and modifies it for speech recognition (van den Oord et al., 2016).

2.2.4. WaveNet

WaveNet (van den Oord et al., 2016), first introduced in 2016, is a deep convolutional neural network. The convolutional layers in WaveNet have various dilation factors that allow its receptive field to grow exponentially with depth and cover thousands of timesteps. Figure 2.4 represents the architecture of this fully-convolutional neural network.

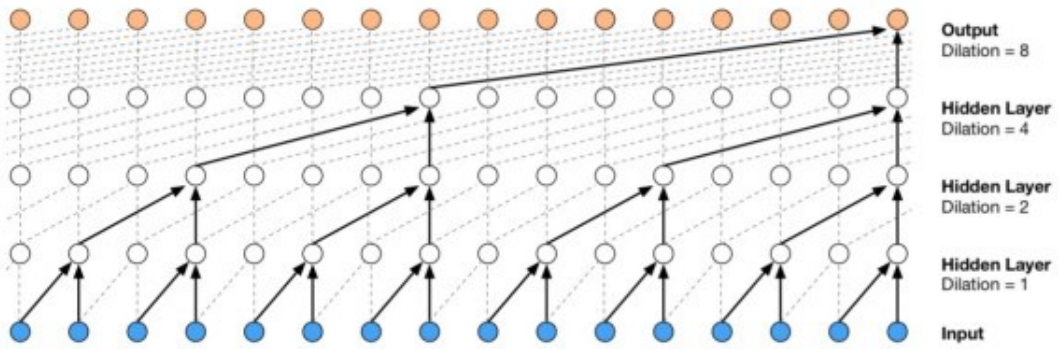

Figure 2.4.: Visualisation of a stack of *dilated* convolutional layers (from van den Oord et al., 2016)

WaveNet enlarges the receptive filed by using *atrous convolution* or *dilated convolution*. Dilating the filter means expanding its size by filling the empty positions with zeros. In practice, no expanded filter is created; instead, the filter elements (the weights) are matched to distant (not adjacent) elements in the input matrix. The distance is determined by the dilation coefficient. In WaveNet, the dilation coefficient is doubled for every layer up to a limit and then repeated. e.g:

`1,2,4,...,512,1,2,4,...512,1,2,4,...,512.`

The activation unit in WaveNet is given in Equation 3.7 below:

$$y = tanh(W_{f,k} * x) \odot \sigma(W_{g,k} * x) \tag{2.7}$$

In this equation, $*$ denotes a convolution operator, $\odot$ denotes an element-wise multiplication operator, $\sigma(.)$ is a sigmoid function, k is the layer index, f and g denote filter and gate respectively, and W is a learnable convolution filter (van den Oord et al., 2016). There are no pooling layers in the network, and the last layer is a softmax layer, optimised to maximise the log likelihood of the output data with respect to the parameters.

WaveNet employs both residual and skip connections throughout the network. As suggested by He et al. (2016) residual and parameterised skip connections speed up convergence and enable training of much deeper models. The architecture of a residual block is shown in the bottom-right corner of Figure 3.1 in the next chapter.

2.3. ASR in Farsi

It is only in the past 20 years that speech recognition for Farsi has been addressed by Iranian researchers (Babaali and Sameti, 2004; Babaali, 2004).

The first continuous speech recogniser for Farsi was built in 1997, with a hybrid architecture of neural networks (a Self-Organising Feature Map, and a Multi-Layer Perceptron). This system recognises Farsi phonemes in the first step, and in the next step the string of phonemes are corrected, segmented and converted to formal text using a non-stochastic method (Sheikhan et al., 1997).

In 2003, Iranian researchers started to develop a large vocabulary speaker-independent Farsi continuous speech recognition system, called Nevisa. The development of this system has since been ongoing; today, Nevisa is the state-of-the-art speech recognition engine for Farsi. Nevisa is an HMM-based system which uses MFCC features of speech signals with some modifications. It features environmental noise robustness techniques, as well as statistical and grammatical language models (Sameti et al., 2011). The reported word error rate of this system, using a trigram language model, is 21.76%.

To date, however, there has been no research on the use of deep learning methods for speech recognition in Farsi. The experiments described in Chapter 4 represent a first step toward a deep learning-based speech recogniser for spontaneous speech in Farsi that can handle registerial/dialectal variations.

A key advantage of CNN approaches to ASR, is that they are theoretically language-agnostic, requiring no hard-coded linguistic rules, but only high-quality training data in a given language. This means that global developments in CNN approaches to ASR can be more easily exploited for other under-resourced languages than earlier methods. The production of a stable, language-agnostic pipeline for CNN-based ASR is therefore timely.

2.4. Chapter summary

In this chapter, I introduced key concepts in contemporary ASR, and situated them in a historical context. CNNs were proposed as a viable approach to ASR for under-resourced languages, Farsi in particular. In the next chapter, I describe currently available datasets and an open-source CNN implementation for ASR, which form major components in my experiments, presented in Chapter 4.

3. Data and Tools

In the previous chapter, I reviewed literature concerning neural-network approaches to ASR. In this chapter, I introduce the data and software tools used in this thesis. This includes `TensorFlow`, Speech-to-Text-WaveNet, and network configurations which remain constant for all experiments presented in Chapter 4.

3.1. Data

Two different speech corpora are used for the experiments in this project in two different languages: English, and Farsi. The English dataset, *CSTR VCTK*, is a read speech corpus (i.e. recordings of written text being read aloud), while the Farsi dataset, *TFarsDat*, is a corpus of telephony spontaneous speech.

VCTK was initially chosen since it is the closest subset of Speech-to-text-WaveNet's original implementation to TFarsDat in terms of size. Another feature of this dataset that makes it suitable for this project, is that like TFarsDat, it contains speech in various dialects. It also makes a good baseline dataset with top-baseline results, by virtue of being a corpus of clean read speech.

The ideal corpus for a dialect normalisation task in Farsi, would be a corpus with clean conversational speech, carefully transcribed in Ketabi/ formal Farsi. TFarsDat has the benefit of being transcribed with both

"Phonetic" and "Phonemic" labels(i.e. in both colloquial and formal language), and it is the only available corpus of conversational Farsi to date. The downside of this dataset is that it contains a lot of noise, as can be expected in a corpus of telephony speech.

The details of each of these corpora follows.

3.1.1. CSTR VCTK Corpus

The CSTR VCTK Corpus (Veaux et al., 2017) includes speech data uttered by 109 native speakers of English with various accents. The ratio of female to male speakers is 62/47 (57% female, 43% male), with each speaker reading approximately 400 sentences aloud. Some of these sentences were selected from a newspaper and some are either from the Rainbow Passage or an elicitation paragraph intended to identify the speaker's accent. The newspaper texts were taken from *The Herald* (Glasgow). Each speaker reads a different set of the newspaper sentences, where each set was selected using a greedy algorithm designed to maximise the contextual and phonetic coverage. The Rainbow Passage and the elicitation paragraph are the same for all speakers. The Rainbow Passage can be found in the "International Dialects of English Archive" webpage (Meier and Muller, 1998). The elicitation paragraph is the same as the one used for "The Speech Accent Archive" (Weinberger and Kunath, 2011).

All speech data was recorded using an identical recording setup: an omni-directional head-mounted microphone (DPA 4035), 96kHz sampling frequency at 24 bits and in a hemi-anechoic chamber of the University of Edinburgh. All recordings were converted into 16 bits, were down-sampled to 48 kHz, and were manually end-pointed.

3.1.2. TFarsDat

TFarsDat (Bijankhan et al., 2003), the Telephone Farsi Speech Database, is an audio collection of phone conversations between pairs of 202 different speakers (Bijankhan et al., 2003). All participants are native speakers of Farsi with differences in age, gender, education level, and dialect. The participants consist of 77 female and 125 male speakers (62% male, 38% female). Audio files in TFarsDat have been recorded with a 16-bit audio card, sampled at 11kHz. A linguist has transcribed and labelled the data at word and phoneme levels using purpose-built software.

Non-linguistic sounds and noises such as movement of the lips, hesitation sounds, line noise, and environmental noises have been labelled as well. Each word-level phonetic label has also been labelled with the equivalent of the spoken word in Ketabi (formal) Farsi. The phonetic labels indicate the actual sounds uttered by the speakers and the phonemic labels are the standard word forms per Ketabi Farsi. An example of a formal–informal word pair in the dataset is `mi]/yim: miy/ym`. The first element is the standard form of the word in Ketabi Farsi, and the second one is the phonetic realisation of the word in the telephone conversation.

Each audio file has been transcribed using a combination of Latin alphabetical and punctuation characters. For example, the character "." (point) denotes the fricative ʃ ("sh") and the character "'" (single quote) denotes the affricate ʧ ("ch"). Table A2 in the Appendix shows the mapping of transcription labels to their corresponding IPA characters. Apart from the words, non-linguistic and paralinguistic noises have also been labeled in the transcriptions. These include breath noise, silence, laughter, and some other noises as shown in Table A3 in the Appendix.

The transcriptions are all stored in XML format, with `Start`, `End`, `Phonetic`, `Phonemic` and `Description` labels. `Start` and `End` point to the data points

at which the word starts and ends in the audio file. `Phonetic` gives the transcription of the word as uttered in the audio file. `Phonemic` contains the equivalent of the spoken word in Ketabi Farsi. Finally, the formal equivalent of uttered word in Persian alphabet is given in `Description`.

3.2. End-to-end speech recognition tools

Progress in ASR research, coupled with increasing computer power and affordability, has made possible the accurate recognition of clean speech spoken in standard dialects of numerous languages. ASR systems have transformed from simple template-based systems to highly complex deep learning-based systems.

A major factor in the improvement of ASR performance has been developments in machine learning. In this section, I review some of the most important machine learning and speech processing tools for ASR, and Speech-to-Text-WaveNet, the implementation of WaveNet for speech recognition, employed in this thesis.

3.2.1. Network configuration: `Speech-to-Text-WaveNet`

The experimental setup of the current work is adapted from Speech-to-Text-WaveNet (Kim and Park, 2016), a TensorFlow implementation of a DeepMind paper on speech synthesis (van den Oord et al., 2016), modified for a speech recognition task. The architecture of Speech-to-Text-WaveNet is shown in Figure 3.1.

Speech-to-Text-WaveNet includes three blocks of dilated convolutional neural networks, each block starting with a dilation coefficient of one, doubled at every layer up to 16. The filter size is seven, and the network is trained with 128 hidden units. The output decoder in Speech-to-Text-

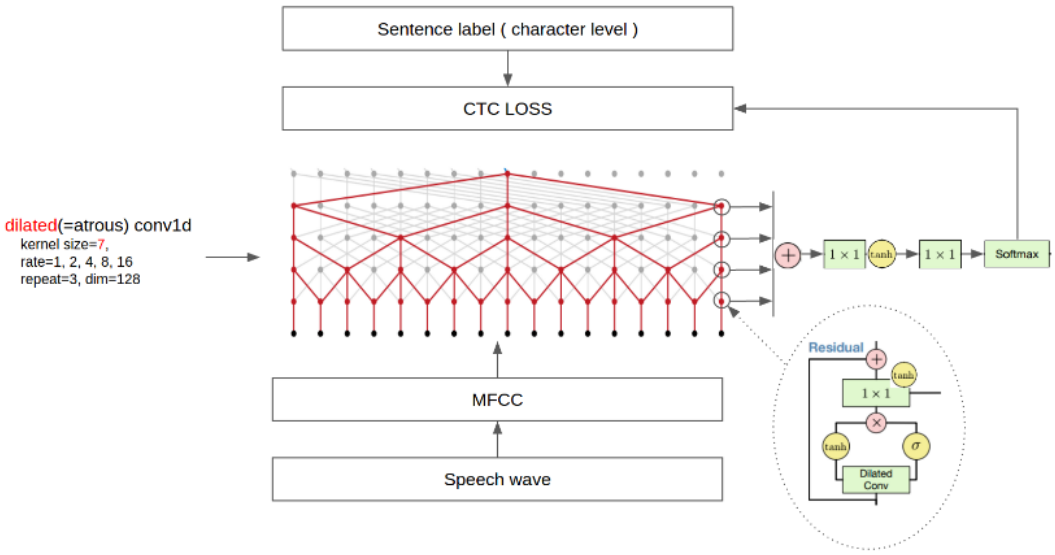

Figure 3.1.: Speech-to-Text-WaveNet pipeline's architecture

WaveNet is a beam search decoder. The same configuration is used for all experiments described in the next chapter.

3.2.2. Feature extraction: `LibROSA`

As shown in Figure 3.1, speech waves are turned into MFCCs, before being fed to the network. Feature extraction is performed using `LibROSA`, an open-source Python package for music and audio analysis. At a high level, `LibROSA` provides implementations of a variety of common functions used throughout the field of music information retrieval and digital signal processing (McFee et al., 2015). The `librosa.feature` module implements a variety of spectral representations, including MFCCs. As mentioned in Section 2.2.1, Mel Frequency Cepstral Coefficients are commonly used to represent audio signals, as they provide a rough model of human frequency perception. `librosa.feature.mfcc` provides a function for extracting MFCC

features. Feature extraction in all experiments described in Chapter 4 is performed using `LibROSA`.

To get the MFCC feature vectors, the audio signal is divided into frames of 25 milliseconds long. This means the frame length for a 16kHz signal is $0.025 \times 16000 = 400$ samples. A step of 10 milliseconds ($0.001 \times 16000 = 160$ samples) is taken to move to the next frame, which allows some overlap in the frames. The first 400 sample frame starts at sample 0, the next 400 sample frame starts at sample 160, etc., until the end of the speech file is reached. The speech files that do not divide into an even number of frames are padded with zeros so that they do. One set of 20 MFCCs is extracted for each frame.

3.2.3. Model implementation: `TensorFlow`

All implementations in this project are performed using `TensorFlow` (Abadi et al., 2016). `TensorFlow` is a library for machine learning across a range of tasks. It was developed by the Google Brain team for internal Google use, but later released as open-source software. One advantage of TensorFlow over other libraries is that it can run on multiple CPUs and GPUs, making the training process much faster than non-parallel training.

`TensorFlow`'s `tf.nn` module supports neural network algorithms. The convolutional neural network is built using this module with configurations specified in Section 3.2.1. `tf.nn.ctc_loss()` is a function that computes the CTC loss, given the output sequence and target labels. The final output decoding is performed using `tf.nn.ctc_beam_search_decoder()`. This function runs beam search with a width of 100, on the logits given the input and returns the final transcription.

3.3. Reference implementation

Speech-to-Text-WaveNet (Kim and Park, 2016) was originally trained on data from three English speech corpora: VCTK (Veaux et al., 2017), LibriSpeech (Panayotov et al., 2015) and TED-LIUM, Release 2 (Rousseau et al., 2012). The total number of sentences in the training set composed of the above three corpora is 240,612. Validation and test sets are available only for LibriSpeech and TED-LIUM; the VCTK Corpus does not have distinct validation and test sets. While the authors of Speech-to-Text-WaveNet have not reported the results of the experiment in terms of label error rate, they have provided a Docker image of their system, facilitating reproduction of their experiments. Testing the Dockerised Speech-to-Text-WaveNet system gave observably high-quality output. Table 3.1 shows examples of pairs of output-target sentences from this implementation.

	Output	Target
1	`he hoped there would be stoo for dinner turnips and charrats and bruzed patatos and fat mutton pieces to be ladled out in th thick peppered flower fatan sauce`	`HE HOPED THERE WOULD BE STEW FOR DINNER TURNIPS AND CARROTS AND BRUISED POTATOES AND FAT MUTTON PIECES TO BE LADLED OUT IN THICK PEPPERED FLOUR FATTENED SAUCE`
2	`numbrt tan fresh nalli is waiting on nou cold nit husband`	`NUMBER TEN FRESH NELLY IS WAITING ON YOU GOOD NIGHT HUSBAND`
3	`o berty and he god in your mind`	`HELLO BERTIE ANY GOOD IN YOUR MIND`

Table 3.1.: Samples of output-target pairs from Speech-to-Text-WaveNet original experiment

As Table 3.1 shows, the overall performance of the system is very good, with the exception of proper nouns, and words with a loose grapheme-phoneme correspondence. This level of accuracy is achieved using hundreds of hours of speech to train the network. TFarsDat, however, contains only 17 hours of Farsi speech (approximately 17,500 sentences). Because

this disparity is certain to seriously affect performance of the system, the first experiment presented in the following chapter involves building a more comparable baseline. To do this, speech-to-Text-WaveNet is retrained using only the VCTK Corpus, a subset of the data used for the original experiment containing approximately 42,600 sentences. This provides a more accurate point of reference for the second and third experiments, which measure the performance of the Farsi ASR system trained on TFarsDat.

3.4. Chapter summary

In this chapter, I described data and software to be used in three experiments, each using CNNs for ASR. These experiments are presented in the next chapter.

4. Experiments

This chapter details three distinct experiments, all centred on CNN-based ASR. The first experiment, described in Section 4.2, is a replication of Speech-to-Text-WaveNet experiment on a subset of the original training data, the VCTK Corpus. The purpose of this experiment is to build a comparable baseline for the experiments to follow.

The second experiment, described in Section 4.3, implements the same architecture on TFarsDat a corpus of Farsi telephone conversations, to produce actual transcriptions of the audio files. This experiment is conducted to answer the first research question, concerning the identification of major challenges in implementing a deep-learning-based pipeline for Farsi ASR.

The third experiment, described in Section 4.4, extends the DNN approach to a dialect normalisation task, in an attempt to answer the second research question, which concerns the extent to which Speech-to-Text-WaveNet has utility for a dialect normalisation task. In this experiment, the network is trained to transcribe conversational speech files with Ketabi/formal Farsi.

4.1. Experimental setup

All three experiments follow the same general procedure:

Data preparation: sentence-level transcriptions are transformed into CSV files, and mapped into audio files with start and end timestamps.

Preprocessing: audio files and their labels are preprocessed before being fed to the network. More specifically, files are transformed into MFCC feature vectors, which are much smaller, but which contain the most important information of the audio signals. Alphabetical characters are transposed into corresponding integers. The alphabet consists of all the graphemes/phonemes used in the dataset plus space.

Network training: the neural network is fed with MFCC feature vectors as inputs and their corresponding labels as outputs. Network configurations are identical to Speech-to-Text-WaveNet described in Section 3.2.1, and fixed for all experiments. Training is run on two GeForce GTX TITAN X GPUs.

Model selection: CTC loss on validation set is calculated every five epochs to choose the best model. The model is then taken from the epoch at which the lowest CTC loss occurs.

Speech Recognition: Audio files in the test set are turned into MFCC feature vectors and fed to the network to get predicted transcriptions, using parameters in the chosen model.

Evaluation: the outputs of the system are evaluated using *Label Error Rate (LER)* at the level of phoneme and word.

4.2. Experiment 1: baseline

The first experiment aims to test performance of the DNN-based approach for end-to-end speech recognition on smaller datasets, in order to build an appropriate baseline for later experiments using TFarsDat. For this task, a model is trained using the VCTK Corpus, an English read speech corpus.

4.2.1. Dataset: VCTK Corpus

The VCTK Corpus contains a total number of 42,583 audio files (one sentence per audio file), uttered by 109 speakers. The original corpus does not include validation and test sets, but was divided for this experiment into train, test, and validation subsets using the following procedure:

- Split the data into train and test sets with an 80/20 ratio
- Split the train set into train and validation sets, again with an 80/20 ratio

The size of each subset after dataset division is given in Table 4.1.

	Train	Validation	Test
Speakers	72	17	20
Sentences	27,728	6,500	8,355

Table 4.1.: VCTK Corpus subsets

4.2.2. Preprocessing

The transcribed training data consists of sentences written using normal English spelling and punctuation rules. Before being fed into the neural network, these labels are lowercased and stripped of punctuation marks,

and indexed based on a dictionary that maps letters to integers. This step also involves creating MFCC feature vectors from audio files as described in Section 3.2.2.

4.2.3. Training

For the first experiment, training was run until Epoch 20. Epoch 10, the epoch with minimum CTC loss on the validation set, was selected. Table 4.2 shows CTC loss at each epoch on train, validation and test sets. The decrease of CTC loss on the train set, coupled with increased CTC loss on validation set after Epoch 10, suggests overfitting of the parameters on training data after this epoch.

Epoch	Train	Validation	Test
5	39.57	53.75	48.49
10	22.11	53.40	48.13
15	16.83	57.25	51.84
20	8.90	62.48	56.29

Table 4.2.: VCTK Corpus CTC loss

The evolution of the transcriptions given by the network at each epoch on data from train and test sets is shown in Tables 4.3 and 4.4.

Epoch	Sample A	Sample B
5	`he lals t be ecatins`	`wel it wonet`
10	`he loves to bie oecasions`	`wel it wont`
15	`he loves the big occpasions`	`well it wont`
20	`he wovsthe big repagions`	`wellh it wonet`
Target	`he loves the big occasions`	`well it wont`

Table 4.3.: Transcriptions of two samples from training data given by the network at different epochs

Epoch	Sample A	Sample B
5	se rederis had ampel an wor realis ic paslect	ois ar l pe ble hant
10	toeendher is now empee ang ore realis tic pasment	is oe l cpeoln hant
15	swerander s naw empeemag wore realis tic prasmet	is here l peoiln hant
20	sobenderis an thempep ll no ir realiystic pasnect	its awl we le han
Target	Surrender is not an appealing or realistic prospect	Is that what people want

Table 4.4.: Transcriptions of two samples from test data given by the network at different epochs

4.3. Experiment 2: DNN-based ASR for Farsi

This experiment tests the replicability of the current pipeline for ASR in Farsi. The experiment is structurally identical to Experiment 1, but with TFarsDat, rather than VCTK, used as training data.

4.3.1. Dataset: TFarsDat

The same data division procedure as in Experiment 1 (Section 4.2) was used to divide TFarsDat into train, validation and test sets. The division was done by file. Accordingly, because files contain variable numbers of sentences, the ratio of sentences in each set differs fractionally from the ratios in the previous experiment.

After dividing the dataset, the train set consisted of 39 audio files, with a total duration of about ten hours of speech. The test set consisted of 15 audio files, with a total duration of about five hours of speech. The validation set consisted of eight audio files, with a total duration of about 145 minutes of speech. Information on number of sentences and speakers in each subset is given in Table 4.5.

	Training	Validation	Test
Speakers	78	16	28
Sentences	10,875	2,908	3,687

Table 4.5.: TFarsDat Corpus subsets for Experiment 2

4.3.2. Data preparation

A key difference between the VCTK Corpus and TFarsDat is that VCTK transcriptions are graphological, while TFarsDat transcriptions are phonetic. To represent the phonemic inventory of Farsi, TFarsDat uses an IPA-like schema, with punctuation symbols standing in for non-ASCII characters. Punctuation marks, therefore, do not mark clausal or sentence boundaries. Because the corpus is made of telephony conversational speech, it is rich in speaker overlap, incomplete sentences, backchannels, and the like. For these reasons, sentence boundaries in the transcriptions are often not clearly marked. This is a second important difference between VCTK and TFarsDat.

For this experiment, the target labels were split at points of silence and laughter, in order to (roughly) approximate sentence boundaries. After being used to split the input data, silence and laughter labels are removed, and skipped in the audio files. However, other transcribed noises, such as lip sounds and line noises, remain in the data, since test data also includes these noises.

4.3.3. Preprocessing

In the preprocessing step, MFCC features were extracted with the same details (frame rate = 25 milliseconds, frame step = 10 milliseconds) using `LibROSA` (see Section 3.2.2 for further information on the extraction process). Since the default sampling rate in `LibROSA` is 16kHz, the audio

files were re-sampled from 11kHz to 16kHz, before feature extraction. The labels are also converted into indices using a list of all phonemes in the alphabet, plus whitespace.

4.3.4. Training

The network was trained until Epoch 25, and the epoch with minimum CTC loss on the validation set, epoch 20, selected. The CTC loss at each epoch on different subsets is shown in Table 4.6. Results from this experiment are presented in Section 4.5.

Epoch	Train	Validation	Test
5	109.19	102.79	107.63
10	95.56	94.06	103.92
15	87.63	90.53	101.44
20	76.94	87.87	97.07
25	72.16	88.28	99.50

Table 4.6.: TFarsDat CTC loss for a normal transcription task

4.4. Experiment 3: dialect normalisation

The final experiment is an attempt to answer the second research question, concerning the plausibility of transcribing conversational speech as formal language in Farsi. The dataset used in this experiment, and its division details are identical to Experiment 2 (see Section 4.3).

4.4.1. Data preparation and preprocessing

Data preparation for this experiment follows the same methodology as in Experiment 2. The only difference is in the labels in the output layer of the neural network. As explained before, the audio files in TFarsDat

are labelled with both `Phonetic` labels, transcriptions of uttered word, and `Phonemic` labels, the Ketabi variants of the uttered words. In this experiment Sound waves are mapped into the `Phonemic` labels, which are equivalents of the spoken words in Formal Farsi, not words as they are uttered.

Preprocessing follows the procedure described in Sections 4.3.2 and 4.3.3. The number of sentences and speakers in each subset is identical to previous experience, shown in Table 4.5.

4.4.2. Training

Training is done until Epoch 35. The epoch with minimum CTC loss on the validation set is chosen, which is epoch 30 for this experiment. The CTC loss at each epoch for train, validation and test sets is shown in Table 4.7.

Epoch	Train	Validation	Test
5	127.52	122.67	109.14
10	110.27	109.49	97.49
15	98.12	103.31	91.06
20	85.41	96.27	85.18
25	77.88	95.85	85.16
30	70.58	94.67	84.22
35	63.64	96.58	86.19

Table 4.7.: TFarsDat CTC loss for formal transcription of conversational speech

Results from this experiment are also presented in Section 4.5, alongside results from the other two experiments.

4.5. Evaluation and results

The general difficulty of measuring performance of a speech recognition system lies in the fact that the recognised word sequence can have a diff-

erent length from the target word sequence. Each recognised word can in turn be slightly different from the target word as well, which makes evaluation even more difficult, especially in case of an end-to-end speech recognition system, which doesn't have a lexicon from which words can be selected.

Graves et al. (2006) introduced *Label Error rate (LER)* as a measure for evaluation of temporal classifiers. For a classifier h, LER is defined as the mean normalised edit distance between its classifications and the targets on test set S', disjoint from training set S:

$$LER(h, |S'|) = \frac{1}{|S'|} \sum_{(x,z) \in S'} \frac{ED(h(x), z)}{|z|} \tag{4.1}$$

Where *ED(p,q)* is the minimum number of insertions, deletions and substitutions that would be required to turn p into q, $|z|$ is the length of target sequence z, and $|S'|$ is the length of test set S'.

Label error rate can be defined according to the type of label in question (word, phoneme, etc.). *Word Error Rate (WER)* and *Phoneme Error Rate (PER)* are sufficient measures for the purpose of this thesis.

Table 4.8 shows the evaluation results for each experiment, with the amount of training data used to train them. To calculate the edit distance between two strings, the open-source Python library `editdistance` is used (Tanaka, 2013). This library is an implementation of *Levenshtein distance*, using the algorithm proposed by Hyyrö (2001).

As evident in Table 4.8, accuracy of the system relatively low in case of Experiment 2 and Experiment 3: For Experiment 2, for example, only 98 per cent of words were correctly transcribed. Given an average sentence length of 10, this means that only 0.2 words per spoken sentence are correctly handled. WER gets even lower in Experiment 3. Possible reasons

Experiment	No. sentences	PER	WER
1	27,728	35%	77%
2	10,875	73%	98%
3	10,875	78%	99%

Table 4.8.: Label Error Rate (LER) and Word Error Rate (WER) on different experiments

behind the low performance in these two experiments are provided in the next chapter.

4.6. Chapter summary

In this chapter, I have detailed three experiments, each concerning deep learning-approaches to ASR. The first experiment is a replication of the system on an English dataset. Experiments 2 and 3 however, are conducted with Farsi as the target language. The experiments are evaluated using label error rate and the results are presented.

The next chapter combines a more detailed discussion of these results with critical reflection on shortcomings of the experiments, suggestions for further research, and the implications of the thesis for ASR researchers, especially in the context of Farsi and other under-resourced languages.

5. Discussion, future work, and conclusion

In the previous chapter, I described three CNN ASR experiments on English and Farsi data. In this chapter, I discuss the findings of these experiments, identifying shortcomings of the approach, implications of the work, and potential areas for further research. A brief summary and conclusion follows.

5.1. Discussion

As Table 4.8 in the previous chapter shows, the experiment with the English corpus resulted in rather low label error rates for a system trained on 27,723 sentences. The gap between WER and PER in this experiment can be understood as a result of the fact that target labels in this experiment are written in standard-language English words and English has a rather loose grapheme-phoneme correspondence. This means for the system to learn the correct spellings of words, a much bigger training data is needed, with multiple occurrences of words that do not have a one-to-one phoneme-grapheme mapping. As an alternative, adding a language model to the system could compensate for the lack of training data, and yield a lower word error rate.

This is a general problem for grapheme-based ASR, even when trained on thousands of hours of speech. Hannun et al. (2014) report a WER of 16.0% for an RNN-based ASR system, trained on 2300 hours of speech (up to three million utterances). They argue that errors tend to be phonetically plausible renderings of English words, that mostly occur on words that rarely or never appear in the training set.

Good examples of this problem area in the ASR process, even when trained on big data, can be seen throughout Table 3.1: `charrats and bruzed potatoes` (target: `carrots and bruised potatoes`). `charrats` is indeed a predictable rendering of `carrots`, based on the fact that word-initial `char` is often pronounced identically (*character*, *charisma*). The unstressed schwa in potatoes, similarly, poses a difficult issue, as it could be orthographically represented with any vowel without an obvious change in pronunciation for most English dialects (`pitatoes`, `putatoes`). Finally, for `fatan sauce` (*fattened sauce*), we can see that a tendency to elide constant clusters in spoken English (here, `nds`) is likely to lead to a pronunciation not dissimilar to `fatan sauce`. Other challenges include idiosyncratic vowel orthography (`bruzed` is perhaps more logical a spelling than *bruised*), and silent consonant phonemes (leading to `nit` being predicted instead of *night*).

The first experiment with the Farsi dataset, also a conventional transcription task, brought about higher error rates when tested on unseen data. The outputs of this system are generally not comprehensible. At the same time, they do contain strings of characters that are identical to words in Farsi, without any changes, or with a small edit distance. A close study of the outputs of the system in Experiments 2 and 3 shows a relationship between the instances of correctly labelled words with the most frequent words in the training data. Some of the observed instances are `xeyli` (English: *very*), `.om/` (the plural, formal, singular second person pronoun), `xub` (*good*, or *well*), `masalan` (*for example*), and `xod/` (*god*). Each

of these well-predicted words are among the top 50 most frequent words in the train set (see Table A4 in the Appendix). This relationship between frequency of words in training data and correct labelling points toward a need for higher quantities of training data, since the most accurately transcribed tokens are the ones observed many times in the train set.

The final experiment, also using the Farsi dataset, was more exploratory. In this experiment, the goal was to get the recogniser to transcribe conversational speech with registerial and dialectal differences, in a certain mode of language (namely, Ketabi Farsi), which is the standard form of the official language of Iran as used in newspapers. The final result of this experiment is very close to that of the normal transcription task (Experiment 2). The resemblance of the results of these two experiments implies the plausability of transcribing conversational Farsi in Ketabi Farsi using the proposed method. If the results of Experiment 2 were to be improved by utilisation of a larger and/or higher-quality dataset in conversational Farsi, the attempt to transcribe audio files in Ketabi Farsi would undoubtedly have been more successful too.

5.1.1. Limitations of the experiments

With a machine learning algorithm at the core of the methodology, the biggest limitation during the course of this project were issues related to available training data. Recognising features and learning to classify them automatically requires very large amounts of high-quality training material, even for the simplest of problems. Entering the domain of speech, the deep learning algorithm is dealing with rather complex data inputs, and this complexity can only be compensated for with abundance of data.

Training data: quality and quantity

Data limitations in the case of this study can be addressed both in terms of quality and quantity.

VCTK and TFarsDat are two datasets with different natures: one is a read speech corpus and the other is a telephony spontaneous speech corpus. The read speech data is inherently cleaner and much less noisy than the telephony data. The noise in the telephony dataset includes, but is not limited to, line and background noise. The ASR system used in this study has not been designed to take the complications of telephony speech into account. Apart from line noises and labelled noises such as laughter and sounds of hesitation, the problem of speaker overlap, where two or more speakers are speaking at the same time, also presents a serious challenge to the system. These overlaps are transcribed with what both speakers are saying at the same time, and happen quite often during a telephone conversation, creating confusion for the system.

Regarding quantity of training data, it is important to note that TFarsDat is a relatively small dataset, even in comparison with the VCTK Corpus. Most experiments on noise robust speech recognition rely on hundreds of hours of speech, while TFarsDat contains only 17 hours of speech (Sainath et al., 2013; Qian and Woodland, 2016; Maas et al., 2015). The comparison between the English system trained on a very large dataset and a subset thereof (Tables 3.1 and 4.4) shows clearly the relationship between data quantity and system performance.

Input representation

Even though MFCC feature vectors have been commonly used in ASR systems, they still have some limitations, arising from the ways in which they are generated from the input audio. As explained by O'Shaughnessy

(2008), aside from $c0$ and $c1$, cepstral coefficients do not have a meaningful interpretation. $c0$ is the power over all frequency bands and $c1$ is the balance between low and high frequency components within the signal frame. Other cepstral coefficients, on the other hand, contain the detail of the spectrum to discriminate the sounds, without a clear interpretation. How MFCC features react to accents or noise, is therefore unknown (O'Shaughnessy, 2008). Using MFCCs as the numerical representation of sound waves reduces the amount of computational power required for further processing. However, it does not accurately depict all features of the sound waves, instead only approximating these features. With fewer restrictions on the amount of computational power, all processing could be done on raw sound waves, resulting in more accurate predictions.

Another challenge related to input representation is the problem of sentence boundaries in TFarsDat. Earlier, it was explained that punctuation marks are used in target labels to represent phonemes, and thus they do not function to mark clausal and sentence boundaries. This has led to approximation of sentences using laughter and silence labels. Since prosodic features such as stress and intonation manifested on speech waves are most meaningful at the level of sentence, having wrong sentence boundaries could be another source of confusion for the system. The extent to which these inaccuracies impact the performance of the system, should be further studied.

5.2. Future work

Based on experiment results, it is sensible to suggest that a Farsi dataset of the size and quality of the VCTK Corpus would produce a model of equivalent accuracy to that of Experiment 1. While the resources needed to produce such a corpus are not insignificant, the experiments show that

the approach is indeed viable. Moreover, it can easily be reimplemented for new languages, given enough resources.

In terms of data quality, it is apparent that issues such as line and background noise, speaker overlap, and non-lexical content (e.g. back-channelling and non-verbal phatic communication) negatively affect the performance of the system. The extent to which these kinds of issues can be fixed by employing speech denoising techniques and preprocessing is also worthy of further investigation.

Building a specific-purpose speech corpus for a dialect normalisation task is the ideal scenario for getting better results using the current pipeline. This corpus would contain a big number of sentences uttered by speakers of different Farsi dialects, recorded in office environment, and transcribed in Ketabi Farsi.

Aside from data quality and quantity, the key difference between the English and Farsi tasks is that the speech recognition task on English data is grapheme-based, while for Farsi, it is phoneme-based. Future research would do well to compare and contrast the relationship between these two output targets and performance of the CNN.

All experiments in this thesis were performed using fixed network configurations. Future research would also benefit from modifying hyper-parameters such as filter size, number of layers, and dilation coefficients, to determine which configurations led to better overall performance.

5.3. Contributions of this thesis

This thesis presents the first attempt at a deep learning-based ASR pipeline for Farsi. In this approach, convolutional neural networks and state-of-the-art ASR techniques are applied, in order to normalise and standardise dialectal and registerial differences in transcriptions of spoken language.

Despite the fact that the performance of the system in the experiments is far below what would be required for a useful downstream application, analysis of the system output demonstrates that the likely cause of the inaccuracy is the quality and quantity of the available data. Regardless of the accuracy of the system presented here, the experiments nonetheless demonstrate the ability of deep learning pipelines to be applied to arbitrary languages. At the same time, however, the approach taken here highlights the centrality of large amounts of high-quality data for training of the neural network. While the development of such resources may pose a new problem for under-resourced languages, it is sensible to suggest that it is more advantageous than development of traditional language-specific systems.

5.4. Summary and conclusion

In this thesis I presented a convolutional neural network architecture that takes MFCC feature vectors as input and tries to maximise the probability of target labels in its output layer, using CTC loss as its objective function. The TensorFlow implementation of this system is applied on two datasets, with three experiments conducted in order to test the replicability of this method for audio transcription in two different languages: English and Farsi.

Performance of the systems is low in general, and not satisfactory for immediate adoption in a downstream application. However, because the major cause of this result is the size and quality of the available training data, it appears that the methodology itself may be viable for Farsi ASR tasks, provided work is undertaken to build larger and higher-quality training datasets. More broadly, the language-agnostic nature of this workflow means that high-quality ASR systems could be developed for

a number of under-resourced languages, with the only prerequisite being the production of a sizable collection of high-quality audio recordings and accompanying graphological/phonemic representations.

Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A System for Large-Scale Machine Learning. In *OSDI*, volume 16, pages 265–283.

Abdel-Hamid, O., Mohamed, A.-r., Jiang, H., Deng, L., Penn, G., and Yu, D. (2014). Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545.

Abdel-Hamid, O., Mohamed, A.-r., Jiang, H., and Penn, G. (2012). Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4277–4280. IEEE.

Atal, B. S. and Hanauer, S. L. (1971). Speech analysis and synthesis by linear prediction of the speech wave. *The journal of the acoustical society of America*, 50(2B):637–655.

Babaali, B. (2004). *Incorporating pruning techniques for improving the performance of an HMM-based continuous speech recognizer*. PhD thesis, Ms thesis, Sharif University of Technology.

Babaali, B. and Sameti, H. (2004). The Sharif speaker-independent large vocabulary speech recognition system. In *The 2nd Workshop on Information Technology & Its Disciplines (WITID 2004)*, pages 24–26.

Besacier, L., Barnard, E., Karpov, A., and Schultz, T. (2014). Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100.

Bijankhan, M., Sheykhzadegan, J., Roohani, M. R., Zarrintare, R., Ghasemi, S. Z., and Ghasedi, M. E. (2003). TFarsDat-the Telephone Farsi speech Database. In *Eighth European Conference on Speech Communication and Technology*.

Bourlard, H. and Morgan, N. (1993). Continuous speech recognition by connectionist statistical methods. *IEEE Transactions on Neural Networks*, 4(6):893–909.

Chinn, M. D. and Fairlie, R. W. (2007). The determinants of the global digital divide: a cross-country analysis of computer and internet penetration. *Oxford Economic Papers*, 59(1):16–44.

Davis, K., Biddulph, R., and Balashek, S. (1952). Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, 24(6):637–642.

Davis, S. and Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE transactions on acoustics, speech, and signal processing*, 28(4):357–366.

Deng, L. and O'Shaughnessy, D. (2003). *Speech processing: a dynamic and optimization-oriented approach*. CRC Press.

Ferguson, J. et al. (1980). Hidden Markov analysis: an introduction. *Hidden Markov Models for Speech*, 1:8–15.

Forgie, J. W. and Forgie, C. D. (1959). Results obtained from a vowel recognition computer program. *The Journal of the Acoustical Society of America*, 31(11):1480–1489.

Gordon, R. G., Grimes, B. F., et al. (2005). *Ethnologue: Languages of the world*, volume 15. SIL International Dallas, TX.

Graves, A. (2012a). Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*.

Graves, A. (2012b). Supervised sequence labelling. In *Supervised Sequence Labelling with Recurrent Neural Networks*, pages 5–13. Springer.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM.

Graves, A. and Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1764–1772.

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., et al. (2014). Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hermansky, H. (1990). Perceptual linear predictive (PLP) analysis of speech. *the Journal of the Acoustical Society of America*, 87(4):1738–1752.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural

networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

Hyyrö, H. (2001). Explaining and extending the bit-parallel approximate string matching algorithm of Myers. Technical report, Technical Report A-2001-10, Dept. of Computer and Information Sciences, University of Tampere, Tampere, Finland.

Itakura, F. (1970). A statistical method for estimation of speech spectral density and formant frequency. *IEICE Trans.*, 53(1):35–42.

Jelinek, F., Bahl, L., and Mercer, R. (1975). Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256.

Juang, B.-H. (1985). Maximum-likelihood estimation for mixture multivariate stochastic observations of Markov chains. *AT&T technical journal*, 64(6):1235–1249.

Juang, B.-H. and Rabiner, L. R. (2005). Automatic speech recognition–a brief history of the technology development. *Georgia Institute of Technology. Atlanta Rutgers University and the University of California. Santa Barbara*, 1:67.

Kim, N. and Park, K. (2016). Speech-to-Text-WaveNet. GitHub repository. Available at: `https://github.com/buriburisuri/speech-to-text-wavenet`.

Lawrence, S., Giles, C. L., Tsoi, A. C., and Back, A. D. (1997). Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks*, 8(1):98–113.

LeCun, Y., Huang, F. J., and Bottou, L. (2004). Learning methods for generic object recognition with invariance to pose and lighting. In

Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, volume 2, pages II–104. IEEE.

Lee, C.-H., Rabiner, L. R., Pieraccini, R., and Wilpon, J. G. (1990). Acoustic modeling for large vocabulary speech recognition. *Computer Speech & Language*, 4(2):127–165.

Levinson, S. E., Rabiner, L. R., and Sondhi, M. M. (1983). An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *The Bell System Technical Journal*, 62(4):1035–1074.

Lippmann, R. P. (1988). Neural network classifiers for speech recognition. *Lincoln Laboratory Journal*, 1:107–124.

Lippmann, R. P. (1989). Review of neural networks for speech recognition. *Neural computation*, 1(1):1–38.

Maas, A., Xie, Z., Jurafsky, D., and Ng, A. (2015). Lexicon-free conversational speech recognition with neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 345–354.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*, pages 18–25.

Megerdoomian, K. (2006). Extending a Persian morphological analyzer to blogs. In *Proceedings of the Second Workshop on Persian Language and Computers*. Citeseer.

Meier, P. and Muller, S. (1998). IDEA: International dialects of English archive. *Accessed May*, 17:2005.

Olson, H. F. and Belar, H. (1956). Phonetic typewriter. *The Journal of the Acoustical Society of America*, 28(6):1072–1081.

O'Shaughnessy, D. (2008). Automatic speech recognition: History, methods and challenges. *Pattern Recognition*, 41(10):2965–2979.

Palaz, D., Collobert, R., et al. (2015). Analysis of CNN-based speech recognition system using raw speech as input. Technical report, Idiap.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). Librispeech: an ASR corpus based on public domain audio books. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5206–5210. IEEE.

Pearce, D. and Picone, J. (2002). Aurora working group: DSR front end LVCSR evaluation au/384/02. *Inst. for Signal & Inform. Process., Mississippi State Univ., Tech. Rep*.

Qian, Y. and Woodland, P. C. (2016). Very deep convolutional neural networks for robust speech recognition. In *Spoken Language Technology Workshop (SLT), 2016 IEEE*, pages 481–488. IEEE.

Rousseau, A., Deléglise, P., and Esteve, Y. (2012). TED-LIUM: an Automatic Speech Recognition dedicated corpus. In *LREC*, pages 125–129.

Sainath, T. N., Kingsbury, B., Saon, G., Soltau, H., Mohamed, A.-r., Dahl, G., and Ramabhadran, B. (2015). Deep convolutional neural networks for large-scale speech tasks. *Neural Networks*, 64:39–48.

Sainath, T. N., Mohamed, A.-r., Kingsbury, B., and Ramabhadran, B. (2013). Deep convolutional neural networks for LVCSR. In *Acoustics,*

Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, pages 8614–8618. IEEE.

Sakai, T. and Doshita, S. (1962). The Phonetic Typewriter. In *IFIP Congress*, volume 445, page 449.

Sameti, H., Veisi, H., Bahrani, M., Babaali, B., and Hosseinzadeh, K. (2011). A large vocabulary continuous speech recognition system for Persian language. *EURASIP Journal on Audio, Speech, and Music Processing*, 2011(1):6.

Sheikhan, M., Tebyani, M., and Lotfizad, M. (1997). Continuous speech recognition and syntactic processing in Iranian Farsi language. *International Journal of Speech Technology*, 1(2):135–141.

Song, W. and Cai, J. (2015). End-to-end deep neural network for automatic speech recognition.

Tanaka, H. (2013). editdistance. GitHub repository. Available at: `https://github.com/aflc/editdistance`.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *CoRR abs/1609.03499*.

Veaux, C., Yamagishi, J., MacDonald, K., et al. (2017). CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit.

Velichko, V. and Zagoruyko, N. (1970). Automatic recognition of 200 words. *International Journal of Man-Machine Studies*, 2(3):223–234.

Weinberger, S. H. and Kunath, S. A. (2011). The Speech Accent Archive: towards a typology of English accents. *Language and Computers*, 73(1):265–281.

Wilpon, J. G., Rabiner, L. R., Lee, C.-H., and Goldman, E. (1990). Automatic recognition of keywords in unconstrained speech using hidden Markov models. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(11):1870–1878.

Appendices

IPA symbol	Orthographic symbols	Phonetic description
i	ـیـ ، ی	High front vowel
e	ـــــ ، ا، ـه	Mid front vowel
a	ـــــَ ، ا	Low Stint vowel
u	و	High back vowel
o	ـــــُ ، و	Mid back vowel
ɒ	ا ، آ	Low back vowel
b	بـ ، ب	Voiced bilabial plosive
p	پـ ، پ	Unvoiced bilabial plosive
d	د	Voiced dental plosive
t	تـ ، ت ، ط	Unvoiced dental plosive
g	گـ ، گ / گاری	Voiced velar plosive
k	کـ ، ک / کار	Unvoiced velar plosive
G	قـ ، ق ، غـ ، غ	Voiced uvular plosive
ʔ	ا ، عـ ، ع ، ئـ ، ء ، ـع	Glottal stop
ʤ	جـ ، ج	Voiced alveopalatal affricate
ʧ	چـ ، چ	Unvoiced alveopalatal affricate
v	و	Voiced labiodental fricative
f	فـ ، ف	Unvoiced labiodental fricative
z	ذ ، ز ، ضـ ، ض ، ظ	Voiced alveolar fricative
s	ثـ ، ث ، سـ ، س ، صـ ، ص	Unvoiced alveolar fricative
ʒ	ژ	Voiced alveopalatal fricative
ʃ	شـ ، ش	Unvoiced alveopalatal fricative
x	خـ ، خ	Unvoiced uvular fricative
h	حـ ، ح ، هـ ، ه، ـه	Unvoiced glottal fricative
m	مـ ، م	Bilabial nasal
n	نـ ، ن	Alveolar nasal
r	ر	Alveolar trill
l	لـ ، ل	Alveolar lateral
j	یـ ، ی	Palatal glide

Table A1.: Farsi alphabet with IPA symbols, orthographic symbols, and phonetic descriptions

label	Description
i	hight front vowel
e	mid front vowel
a	low front vowel
u	high back vowel
o	mid back vowel
/	low back vowel
b	voiced bilabial plosive
p	unvoiced bilabial plosive
d	voiced dental plosive
t	unvoiced dental plosive
;	voiced palatal plosive
g	voiced velar plosive
c	unvoiced palatal plosive
k	unvoiced velar plosive
q	voiced uvular plosive
]	glottal stop
’	voiced alveopalatal affricate
,	unvoiced alveopalatal affricate
v	voiced labiodental fricative
f	unvoiced labiodental fricative
z	voiced alveolar fricative
s	unvoiced alveolar fricative
[	voiced alveopalatal fricative
.	unvoiced alveopalatal fricative
x	unvoiced uvular fricative
h	unvoiced glottal fricative
m	bilabial nasal
n	alveolar nasal
r	alveolar trill
l	alveolar lateral
y	palatal glide

Table A2.: TFarsDat characters

Paralinguistic label	Description
ls	lip sound
br	breath sound
uh	pause
cog	cough
bn	background noise
hes	hesitation sound
ns	non-speech sound
ln	line noise
def	incomplete word
deff	unclear word
sil	silence
laugh	laughter

Table A3.: Different types of noise in TFarsDat Corpus

Rank	Token	Frequency	Rank	Token	Frequency
1	ke	5,121	51	nemid/nam	368
2	]in	3,298	52	mikonad	358
3	va	3,122	53	mikonam	355
4	r/	2,799	54	.ode	353
5	ham	2,676	55	hame	340
6	]ast	2,670	56	d/rand	337
7	be	2,362	57	’iz	310
8	yek	2,330	58	fekr	307
9	bale	1,958	59	do	307
10	]az	1,957	60	tu+e	302
11	masalan	1,710	61	be.avad	300
12	m/	1,528	62	nist	291
13	man	1,418	63	d/.te	281
14	xob	1,155	64	bi.tar	265
15	digar	1,128	65	’on	258
16	xeyli	1,119	66	ham/n	249
17	.om/	1,113	67	nazar+e	245
18	tu	1,106	68	’izi	239
19	h/l/	1,076	69	yeki	239
20	]/n	1,024	70	]/dam	237
21	dar	976	71	]/re	230
22	]inh/	944	72	bude	230
23	b/	902	73	d/rim	222
24	]al]/n	742	74	]/n,/	220
25	dorost	693	75	b/z	218
26	’e	668	76	xode.	218
27	vali	648	77	s/l	215
28	b/.ad	626	78	ziy/d	204
29	mi.avad	587	79	konand	200
30	ya]ni	573	80	faqat	200
31	y/	567	81	]/nh/	199
32	d/rad	542	82	xod+e	198
33	bar/ye	536	83	vaqti	198
34	]agar	535	84	miguyand	198
35	ba]d	524	85	]/q/	198
36	hast	519	86	mi]/yad	193
37	b/yad	519	87	konad	189
38	bud	515	88	miguyam	189
39	nm	515	89	beharh/l	185
40	na	491	90	]albatte	180
41	t/	488	91	]in,/	179
42	hamin	480	92	tehr/n	175
43	]aslan	475	93	./yad	174
44	k/r	450	94	]en./]all/h	172
45	xub	433	95	xod/	170
46	hastand	422	96	mesl+e	167
47	v/qe]an	418	97	mardom	167
48	be]estel/h	406	98	ba’’eh/	165
49	har	389	99	karde	164
50	mikonand	379	100	’and	163

Table A4.: Most common tokens in TFassDat training data