Chapter 1

Introduction

Automatic Speech Recognition is a subset of Machine Translation that takes a sequence of raw audio information and translates or matches it against the most likely sequence of text as would be interpreted by a human language expert. In this thesis, Automatic Speech Recognition will also be referred to as ASR or speech recognition for short.

It can be argued that while ASR has achieved excellent performance in specific applications, much is left to be desired for general purpose speech recognition. While commercial applications like Google voice search and Apple Siri gives evidence that this gap is closing, there is still yet other areas within this research space that speech recognition task is very much an unsolved problem.

It is estimated that there are close to 7000 human languages in the world (Besacier et al., 2014) and yet for only a fraction of this number have there been efforts made towards ASR. The level of ASR accuracy that have been so far achieved are based on large quantities of speech data and other linguistic resources used to train models for ASR. These models which depend largely on pattern recognition techniques degrade tremendously when applied to different languages other than the languages that they were trained or designed for. In addition, due to the fact that collection of sufficient amounts of linguistic resources required to create accurate models for ASR are particularly laborious and time consuming to collect sometimes extending to decades, it is therefore wise to consider alternative approaches towards developing ASR systems for languages lacking the resources required to build such systems using existing mechanisms.

REFERENCE OR EVIDENCE

A LONG SENTENCE, TRY AND SPLIT IT IN TWO.

1.1 ASR As a Machine Learning problem

Automatic speech recognition can be put into a class of machine learning problems described as sequence pattern recognition because an ASR attempts to discriminate a pattern from the sequence of speech utterances.

One immediate problem realised with this definition leads us to discuss statistical speech models that address how to handle the problem described in the following paragraph.

Speech is a complex phenomena that begins as a cognitive process and ends up as a physical process. The process of automatic speech recognition attempts to reverse engineer the steps back from the physical process to the cognitive process giving rise to latent variables or mismatched data or loss of information from interpreting speech information from one physiological layer to the next.

It has been acknowledged in the research community (Deng and Li, 2013, Watanabe and Chien, 2015) that work being done in Machine Learning has enhanced the research of automatic speech recognition. Similarly any progress made in ASR usually constitutes a contribution to enhances made in the machine learning field. This also is an attribution to the fact that speech recognition is a sequence pattern recognition problem. Therefore techniques within speech recognition could be applied generally to sequence pattern recognition problems.

The two main approaches to machine learning problems historically involve two methods rooted in statistical science. These approaches are the generative and discriminitative models. From a computing science perspective, the generative approach is a brute-force approach while the discriminative model uses a rather heuristic approach to machine learning. This chapter derives the basic definitions of these two approaches in order to establish the motivation for the proposed models used in this research for low resource speech recognition, as well as introduces the Wakirike language as the motivating language case study.

1.2 Generative-Discriminative Speech Models disambiguation

In the next chapter, the Hidden Markov Model (HMM) is examined as a powerful and major driver behind generative modeling of sequential data like speech. Generative models are data-sensitive models because they are derived from the data by accumulating as many different features which can be seen and make generalisations based on the features seen. The discriminative model, on the other hand, has a heuristic approach to make classification. Rather than using features of the data directly, the discriminative method attempts to characterise the data into features. It is possible to conclude that the generative approach uses a bottom-to-top strategy starting with the fundamental structures to determine the overall structure, while the discriminative method uses a top-to-bottom approach starting with the big picture and then drilling down to discover the fundamental structures.

Ultimately, the generative models for machine learning learning can be interpreted mathematically as a joint distribution that produces the highest likelihood of outputs and inputs based on a predefined decision function. The outputs for speech recognition being the sequence of words and the inputs for speech being the audio wave form or equivalent speech sequence.

$$d_y(\mathbf{x};\lambda) = p(\mathbf{x}, y;\lambda) = p(\mathbf{x}|y;\lambda)p(y;\lambda) \tag{1.1}$$

where $d_y(\mathbf{x};\lambda)$ is the decision function of y for data labels $\mathbf{x}$. This joint probability expression given as $p(\mathbf{x}|y;\lambda)$ can also be expressed as the conditional probability product in equation (1.1). In this equation, λ predefines the nature of the distribution (Deng and Li, 2013) referred to as model parameters.

Similarly, machine learning discriminative models are described mathematically as the conditional probability defined by the generic decision function below:

$$d_y(\mathbf{x};\lambda) = p(y|\mathbf{x};\lambda) \tag{1.2}$$

It is clearly seen that the discriminative paradigm is a much simpler and a more straight forward paradigm and indeed is the chosen paradigm for this study. How-

ever, what the discriminative model gains in discriminative simplicity it loses in model parameter estimation (λ) in equation (1.1) and (1.2). As this research investigates, ~~the~~ although the generative process is able to generate arbitrary outputs from learned inputs, its major draw back is the direct dependence on the training data from which the model parameters are learned. Specific characteristics of various machine learning models are reserved for later chapters albeit the heuristic nature of the discriminative approach, not directly dependent on the training data, gains over the generative approach being able to better compensate the latent variables. In the case of speech data information is lost in training data due to the physiologic transformations mentioned in section 1.1. This rationale is reinforced from the notion of deep learning defined in Deng et al. (2014) as an attempt to learn patterns from data at multiple levels of abstraction. Thus while shallow machine learning models like hidden Markov models (HMMs) define latent variables for fixed layers of abstraction, deep machine learning models handle hidden/latent information for arbitrary layers of abstraction determined heuristically. As deep learning are typically implemented using deep neural networks, this work applies deep recurrent neural networks as an end-to-end discriminative classifier, to speech recognition. This is a so called "an end-to-end model" because it adopts the top-to-bottom machine learning approach. Unlike the typical generative classifiers that require sub-word acoustic models, the end-to-end models develop algorithms at higher levels of abstraction as well as the lower levels of abstraction. In the case of the deep-speech model (Hannun et al., 2014b) utilised in this research, the levels of abstraction include sentence/phrase, words and character discrimination. A second advantage of the end-to-end model is that because the traditional generative models require various stages of modeling including an acoustic, language and lexicon, the end-to-end discriminating multiple levels of abstractions simultaneously only requires a single stage process, greatly reducing the amount of resources required for speech recognition. From a low resource language perspective this is an attractive feature meaning that the model can be learned from an acoustic only source without the need of an acoustic model or a phonetic dictionary. In theory this deep learning technique is sufficient in itself without a language model. However, applying a language model was found to serve as a correction factor further improving recognition results (Hannun et al., 2014a).

1.3 Low Resource Languages

A second challenge observed in complex machine learning models for both generative as well as discriminative learning models is the data intensive nature required for robust classification models. Saon et al. (2015) recommends around 2000 hours of transcribed speech data for a robust speech recognition system. As we cover in the next chapter, for new languages for which are low in training data such as transcribed speech, there are various strategies devised for low resource speech recognition. Besacier et al. (2014) outlines various matrices for bench-marking low resource languages. From the generative speech model interest perspective, reference is made to languages having less than ideal data in transcribed speech, phonetic dictionary and a text corpus for language modelling. For end-to-end speech recognition models interests, the data relevant for low resource evaluation is the transcribed speech and a text corpus for language modelling. It is worth noting that it was observed (Besacier et al., 2014) that speaker-base often doesn't affect a language resource status of a language and was often observed that large speaker bases could in fact lack language/speech recognition resources and that some languages having small speaker bases did in fact have sufficient language/ speech recognition resources.

Speech recognition methods looked at in this work was motivated by the Wakirike language discussed in the next section, which is a low resource language by definition. Thus this research looked at low research language modeling for the Wakirike language from a corpus of Wakirike text available for analysis. However, due to the insufficiency of transcribed speech for the Wakirike language, English language was substituted and used as a control variable to study low resource effects of a language when exposed to speech models developed in this work.

1.4 The Wakirike Language

The Wakirike municipality is a fishing community comprising 13 districts in the Niger Delta area of the country of Nigeria in the West African region of the continent of Africa. Wakirike migrants settled at the Okrika mainland between AD860 at the earliest AD1515 Earliest settlers migrated from Central and Western Niger Delta. When the second set of settlers met the first set of settlers they exclaimed "we are

not different" or "Wakirike" (S., 2008). Although the population of the Wakirike community from a 1995 report (Simons and Fennig, 2018) is about 248,000, the speaker base is much less than that. The language is classified as Niger-Congo and Ijoid languages. The writing orthography is Latin and the language status is 5 (developing) (Simons and Fennig, 2018). This means that although the language is not yet an endangered language, it still isn't thriving and it is being passed on to the next generation at a limited rate.

The Wakirike language was the focus for this research. And End-to-end deep neural network language model was built for the Wakirike language based on the availability of the new testament bible printed edition that was available for processing. The corpus utilized for this thesis work ~~was about~~ 9,000 words.

Due to limitations in transcribed speech for the Wakirike language, English was substituted and used for the final speech model. The English language was used as a control variable to measure accuracy of speech recognition for differing spans of speech data being validated against on algorithms developed in this research.

1.5 Thesis outline

The outline of this report follows the development of an end-to-end speech recogniser and develops the theory based on the building blocks of the final system. Chapter two introduces the speech recognition pipeline and the generative speech model. Chapter two outlines the weaknesses in the generative model and describes some of the machine learning techniques applied to improve speech recognition performance.

Various Low speech recognition methods are reviewed and the relevance of this study is also highlighted. Chapter three describes Recurrent neural networks beginning from multi-layer perceptrons and probabilistic sequence models. Specialised recurrent neural networks, long short-term memory (LSTM) networks and the Gated Recurrent Units (GRU) used to develop the language model for the Wakirike language are detailed.

Chapter Four explains the wavelet theorem as well as the deep scattering spectrum. The chapter develops the theory from Fourier transform and details the the significance of using the scattering transform as a feature selection mechanism for

low resource recognition.

Chapters five and six is a description of the models developed by this thesis and details the experiment setup along with the results obtained. Chapters seven is a discussion of the result and chapter 8 are the recommendations for further study.

Chapter 2

Low Resource Speech Models, End-to-end models and the scattering network

The speech recogniser developed in this thesis is based on an end-to-end discriminative deep recurrent neural network. Two models were developed. The first model is a Gated-Recurrent-Unit Recurrent Neural network (GRU-RNN) was used to develop a character-based language model, while the second recurrent neural network is a Bi-Directional Recurrent neural Network (BiRNN) used as an end-to-end speech model capable of generating word sequences based on learned character sequence outputs. This chapter describes the transition from generative speech models to these discriminative end-to-end recurrent neural network models. Low speech recognition strategies are also discussed and the contribution to knowledge gained by using character-based discrimination as well as introducing deep scattering features to the biRNN speech model is brought to light.

2.1 Speech Recognition Overview

Computer speech recognition takes raw audio speech and converts it into a sequence of symbols. This can be considered as an analog to digital conversion as a continuous signal becomes discretised. The way this conversion is done is by breaking up the audio sequence into very small packets referred to as frames and developing

discriminating parameters or features for each frame. Then using the vector of features as input to the speech recogniser.

A statistical formulation (Young et al., 2002) for the speech recogniser follows given that each discretised output word in the audio speech signal is represented as a vector sequence of frame observations defined in the set $\mathbf{O}$ such that

$$\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \ldots, \mathbf{o}_T. \tag{2.1}$$

At each discrete time t, we have an observation $\mathbf{o}_t$, which is, in itself is a vector in R^D. From the conditional probability, it can be formulated that certain word sequences from a finite dictionary are most probable given a sequence of observations. That is:

$$arg\max_t\{P(w_i|\mathbf{O})\} \tag{2.2}$$

As we describe in the next section on speech recognition challenges, there is no straightforward analysis of of $P(w_i|\mathbf{O})$. The divide and conquer strategy therefore employed uses Bayes formulation to simplify the problem. Accordingly, the argument that maximises the probability of an audio sequence given a particular word multiplied by the probability of that word is equivalent to the original posterior probability required to solve the isolated word recognition problem. This is summarised by the following equation

$$P(w_i|\mathbf{O}) = \frac{P(\mathbf{O}|w_i)P(w_i)}{P(\mathbf{O})} \tag{2.3}$$

That is, according to Bayes' rule, the posterior probability is obtained by multiplying a certain likelihood probability by a prior probability. The likelihood in this case, $P(\mathbf{O}|w_i)$, is obtained from a Hidden Markov Model (HMM) parametric model such that rather than estimating the observation densities in the likelihood probability, these are obtained by estimating the parameters of the HMM model. The HMM model explained in the next section gives a statistical representation of the latent variables of speech.

The second parameter in the speech model interpreted from Bayes' formula is prior is the probability a given word. This aspect of the model is the language model which we review in section 2.2.1.

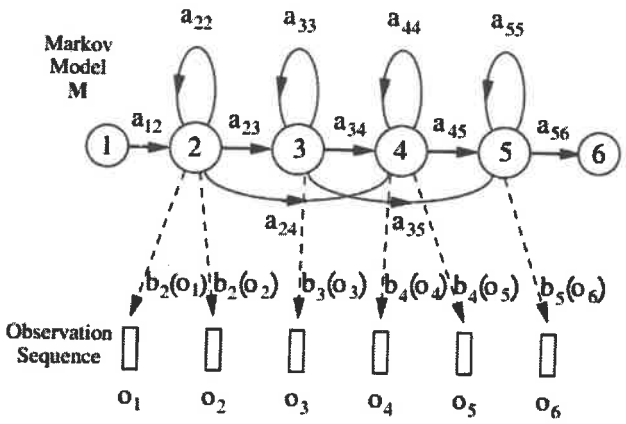


Figure 2.1: HMM Generative Model
Young et al. (2002)

2.1.1 HMM-based Generative speech model

A HMM represents a finite state machine where a process transits a sequence of states from a set of fixed states. The overall sequence of transitions will have a start state, an end state and a finite number of intermediate states all within the set of finite states. ~~For~~ each state transition emits an output observation that represents the current internal state of the system.

In an HMM represented in figure 2.1 there are two important probabilities. The first is the state transition probability given by a_{ij} this is the probability to move from state i to state j. The second probability b_j is the probability that an output probability when a state emits an observation.

Given that X represents the sequence of states transitioned by a process, a HMM, the joint probability of X and the output probabilities given the HMM is given as the following representation:

$$P(\mathbf{O}|M) = \sum_{X} a_{x(0)x(1)} \prod_{t=1}^{T} b_{x(t)}(\mathbf{o}_t) a_{x(t)x(t+1)} \tag{2.4}$$

Generally speaking, the HMM formulation presents 3 distinct challenges. The first is ~~that~~ likelihood of a sequence of observations given in equation 2.4 above. The next two, ~~which we~~ describe later, is the inference and the learning problem. While the inference problem determines the sequence of steps given the emission probabilities, the learning problem determines the HMM parameters, that is the initial transition and emission probabilities of the HMM model.

For the case of the inference problem, the sequence of states can be obtained by determining the sequence of states that maximises the probability of the output sequences.

2.1.2 Challenges of Speech Recognition

The realised symbol is assumed to have a one to one mapping with the segmented raw audio speech. However, the difficulty in computer speech recognition is the fact that there is significant amount of variation in speech that would make it practically intractable to establish a direct mapping from segmented raw speech audio to a sequence of static symbols. The phenomena known as co articulation has it that there are several different symbols having a mapping to a single waveform of speech in addition to several other varying factors including the speaker mood, gender, age, the medium of speech transduction, the room acoustics, et cetera.

Another challenge faced by automated speech recognisers is the fact that the boundaries of the words is not apparent from the raw speech waveform. A third problem that immediately arises from the second is the fact that the words from the speech may not strictly follow the words in the selected vocabulary database. Such occurrence in speech recognition research is referred to as out of vocabulary (OOV) terms. It is reasonable to approach these challenges using a divide and conquer strategy. In this case, the first step in this case would be to create assumption that somehow word boundaries can be determined. This first step in speech recognition is referred to as the isolated word recognition case.

2.1.3 Challenges of low speech recognition

Speech recognition for low resource languages poses another distinct set of challenges. In chapter one, low resource languages were described to be languages lacking in resources required for adequate machine learning of models required for generative speech models. These resources are described basically as a text corpus for language modelling, a phonetic dictionary and transcribed audio speech for acoustic modelling. Figure 2.2, illustrates how resources of required for speech recognition are utilised. It is observed that in addition to the three resources identified other processes are required for the speech decoder to function normally. For example,

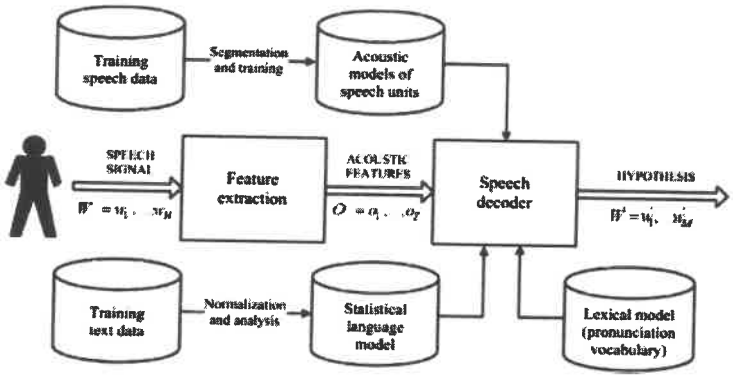


Figure 2.2: Automatic Speech Recognition Pipeline
Besacier et al. (2014)

aligned speech would also need to be segmented into speech utterances to ensure that the computer resources are used conservatively.

In terms of data collection processing Besacier et al. (2014) enumerates the challenges for developing low resource ASR systems to include the fact that phonologies (or language sound systems) differ across languages, word segmentation problems, fuzzy grammatical structures, unwritten languages, lack of native speakers having technical skills and the multidisciplinary nature of ASR constitute impedance to ASR system building.

2.2 Low Resource Speech Recognition

In this system building speech recognition research, the focus was on the development of a language model and and an end-to-end speech model comparable in performance to state of the art speech recognition system consisting of an acoustic model and a language model. Low resource language and acoustic modelling is now reviewed keeping in mind that little work has been done on low-resource end-to-end speech modelling when compared to general end-to-end speech modelling and general speech recognition as a whole.

From an engineering perspective, a practical means of achieving low resource speech modeling from a language rich in resources is through various strategies of the machine learning sub-field of transfer learning.

Transfer learning takes the inner representation of knowledge derived from training an algorithm used from one domain and applying this knowledge in a similar

domain having different set of system parameters. Early work of this nature was for speech recognition is demonstrated in (Vu and Schultz, 2013) where multi-layer perceptrons were used to train multiple languages rich in linguistic resources. In a later section titled speech recognition on a budget, a transfer learning mechanism involving deep neural networks from (Kunze et al., 2017) is described.

2.2.1 Low Resource language modelling

General language modelling is reviewed and then Low resource language modelling is discussed in this section. Recall from the general speech model influenced by Bayes' theorem. The speech recognition model is a product of an acoustic model (likelihood probability) and the language model (prior probability). The development of language models for speech recognition is discussed in Juang and Furui (2000) and Young (1996).

Language modelling formulate rules that predict linguistic events and can be modeled in terms discrete density $P(W)$, where $W = (w_1, w_2, ..., w_L)$ is a word sequence. The density function $P(W)$ assigns a probability to a particular word sequence W. This value determines how likely the word is to appear in an utterance. A sentence with words appearing in a grammatically correct manner is more likely to be spoken than a sentence with words mixed up in an ungrammatical manner, and, therefore, is assigned a higher probability. The order of words therefore reflect the language structure, rules, and convention in a probabilistic way. Statistical language modeling therefore, is an estimate for $P(W)$ from a given set of sentences, or corpus.

The prior probability of a word sequence $\mathbf{w} = w_1, \ldots, w_k$ required in equation (2.2) is given by:

$$P(\mathbf{w}) = \prod_{k=1}^{K} P(w_k | w_{k-1}, \ldots, w_1) \tag{2.5}$$

The N-gram model is formed by conditioning of the word history in equation 2.5. This therefore becomes:

$$P(\mathbf{w}) = \prod_{k=1}^{K} P(w_k | w_{k-1}, w_{k-2}, \ldots, w_{k-N+1}) \tag{2.6}$$

N is typically in the range of 2-4.

N-gram probabilities are estimated from training corpus by counting N-gram occurrences. This is plugged into maximum likelihood (ML) parameter estimate. For example, Given that N=3 then the probability that three words occurred is assuming $C(w_{k-2}w_{k-1}w_k)$ is the number of occurrences of the three words $C(w_{k-2}w_{k-1})$ is the count for $w_{k-2}w_{k-1}w_k$ then

$$P(w_k|w_{k-1}, w_{k-2} \approx \frac{C(w_{k-2}w_{k-1}w_k}{C(w_{k-2}w_{k-1})} \tag{2.7}$$

The major problem with maximum likelihood estimation scheme is data sparsity. This can be tackled by a combination of smoothing techniques involving discounting and backing-off. The alternative approach to robust language modelling is the so-called class based models (Brown et al., 1992, ?) in which data sparsity is not so much an issue. Given that for every word w_k, there is a corresponding class c_k, then,

$$P(\mathbf{w}) \prod_{k=1}^{K} P(w_k|c_k)p(c_k|c_{k-1}, \ldots, c_{k-N+1}) \tag{2.8}$$

In 2003, Bengio et al. (2003) proposed a language model based on neural multilayer perceptrons (MLPs). These MLP language models resort to a distributed representation of all the words in the vocabulary such that the probability function of the word sequences is expressed in terms of these word-level vector representations. The result of the MLP-based language models was found to be, in cases for models with large parameters, performing better than the traditional n-gram models.

Improvements over the MLPs still using neural networks over the next decade include works of Luong et al. (2013), Mikolov et al. (2011), Sutskever et al. (2014), involved the utilisation of deep neural networks for estimating word probabilities in a language model. While a Multi-Layer Perceptron consists of a single hidden layer, in addition to the input and output layers, a deep network, in addition to having several hidden layers are characterised by complex structures that render the architecture beyond the basic feed forward nature where data flows from input to output hence in the RNN architecture we have some feedback neurons as well. Furthermore, the probability distributions in these deep neural networks were either based upon word or sub-word models, this time having representations which also conveyed some level of syntactic or morphological weights to aid in establishing word

relationships. These learned weights are referred to as token or unit embedding.

For the neural network implementations so far seen, a large amount of data is required due to the nature of words to have large vocabularies, even for medium-scale speech recognition applications. Kim et al. (2016) on the other hand took a different approach to language modelling taking advantage of the long-term sequence memory of long-short-term memory cell recurrent neural network (LSTM-RNN) to rather model a language based on characters rather than on words. This greatly reduced the number of parameters involved and therefore the complexity of implementation. This method is particularly of interest to this article and forms the basis of the implementation described in this article due to the low resource constraints imposed when using a character-level language model.

DO YOU MEAN YOUR THESIS OR KIM et al?

Other low resource language modelling strategies employed for the purpose of speech recognition was demonstrated by Xu and Fung (2013). The language model developed in that work was based on phrase-level linguistic mapping from a high resource language to a low resource language using a probabilistic model implemented using a weighted finite state transducer (WFST). This method uses WFST rather than a neural network due to scarcity of training data required to develop a neural network. However, it did not gain from the high non linearity ability of a neural network model to discover hidden patterns in data, being a shallower machine learning architecture.

YOUR THESIS OR XU & FUNG?

The method employed in this report uses a character-based Neural network language model that employs an LSTM network similar to that of Kim et al. (2016) on the Okrika language which is a low resource language, bearing in mind that the character level network will reduce the number of parameters required for training, just enough to develop a working language model for the purpose of speech recognition.

2.2.2 Low Resource Acoustic modelling

Two transfer learning techniques for acoustic modelling investigated by Povey et al. (2011) and Ghoshal et al. (2013) respectively include the sub-space Gaussian mixture models (SGMMs) and the use of pretrained hidden layers of a deep neural network trained multilingually as a means to initialise weights for an unknown language. This second method has been informally referred to as the swap-hat method.

OF LOW RESOURCE MODELLING

Recall that one of the challenges associated with new languages is that phonetic systems differ from one language to another. Transfer learning approaches attempt however to recover patterns common to seemingly disparate systems and model these patterns.

For phonetic systems, based on the premise that sounds are produced by approximate movements and positions of articulators comprising the human speech sound system which is common for all humans. It is possible to model dynamic movement from between various phones as tied state mixture of Gaussians. These dynamic states are modeled using Gaussian mixture models or GMM are also known as senones. Povey et al. (2011) postulated a method to factorize these Gaussian mixtures into a globally shared set of parameters that are not dependent individual HMM states. These factorisations model senones that are not represented in original data and thought to be a representation of the overall acoustic space. While preserving individual HMM states, the decoupling of the shared space and its reuse makes SGMMs a viable candidate for transfer learning of acoustic models for new languages.

The transfer learning procedure proposed in Ghoshal et al. (2013) employed the use of deep neural networks, in particular deep belief networks (Bengio et al., 2007). Deep Belief Networks are pretrained, layer-wise stacked Restricted Boltzmann Machines (RBMs)(Smolensky, 1986). The output of this network trained on senones correspond to HMM context dependent states. However, by decoupling hidden layers from outer and output layers and fine-tuned to a new language, the network is shown to be insensitive to the choice of languages analogous to global parameters of SGMMs. The 7-layer, 2000 neuron per layer network used did not utilise a bottleneck layer corresponding to triphone states trained on MFCC features (Grezl and Fousek, 2008).

2.3 Groundwork for low resource end-to-end speech modelling

The underpinning notion of this work is firstly a departure from the bottom-to-top baggage that comes as a bye-product of the generative process sponsored by the

HMM-based speech models so that we can gain from simplifying the speech pipeline from acoustic, language and phonetic model to just a speech model that approximates the same process. Secondly, the model developed seeks to overcome the data intensity barrier and was seen to achieve measurable results for GRU RNN language models. Therefore adopting the same character-based strategy, this research performed experiments using the character-based bi-directional recurrent neural networks (BiRNN). However, BiRNNs researchers have found them as other deep learning algorithms, as being very data intensiveHannun et al. (2014a). The next paragraphs introduce Deep-speech BiRNNs and the two strategies for tackling the data intensity drawback as related with low resource speech recognition.

2.3.1 Deep speech

THE CENTRE IS IN THE MIDDLE AND CANNOT BE AROUND SOMETHING.

Up until recently, speech recognition research has been centred around improvements of the HMM-based acoustic models. This has included a departure from generative training of HMM to discriminative training (Woodland and Povey, 2000) and the use of neural network precursors to initialise the HMM parameters (Mohamed et al., 2012). Although these discriminative models brought improvements over generative models, being HMM dependent speech models they lacked the end-to-end nature. This means that they were subject to training of acoustic, language and phonetic models. With the introduction of the Connectionist Temporal Classification (CTC) loss function, Graves and Jaitly (2014) finally found a means to end-to-end speech recognition departing from HMM-based speech recognition.

The architecture of the Deep-speech end-to-end speech recognition model Hannun et al. (2014b) follows an end-to-end Bi-directional Recurrent Neural Network (BiRNN) and CTC loss function (Graves et al., 2006). The CTC loss function uses a modified beam search to sum over all possible input-output sequence alignments thereby maximising the likelihood of the output sequence characters.

2.3.2 Speech Recognition on a low budget

In this section, a recent transfer learning speech model model (Kunze et al., 2017) that has some characteristics similar to the speech model developed in this thesis is reviewed. The end-to-end speech model described by Kunze et al. (2017) is based

on that developed by Collobert et al. (2016) and is based on deep convolutional neural networks rather than the Bi-RNN structure proposed by this work. In addition it uses a loss function based on the AutoSegCriterion which is claimed to work competitively with raw audio waveform without any preprocessing. The main strategy for low resource management in their system was the freezing of some layers within the convolutional network layer. The low resource mechanisms used in this work includes the use of a unique scattering network being used as input features for the BiRNN model. The fascinating similarity between the end-to-end BiRNN speech model developed in this work and the transfer learning model in Kunze et al. (2017) is the fact that the scattering network input are equivalent to the output of a light-weight convolutional neural network Hannun et al. (2014b). Therefore the proposed system then approximates a combination of a recurrent neural network as well as a convolution neural network without the overhead of actually training a convolutional neural network (CNN).

Introduction of the unique scattering network is discussed in the next section. It is worthy to note however that Kunze et al. (2017) uses a CNN network only, while (Amodei et al., 2016) uses both RNN and CNN network. The speech model in this thesis uses a BiRNN model ~~in this work~~ combines an RNN model with the scattering layer, which represents a light-weight low resource friendly pseudo enhanced CNN backing. What is meant by pseudo enhanced CNN backing is reserved for the next section, however, therefore, the proposed speech model in this thesis stands to gain from an enhanced but lightweight CNN combined with RNN learning.

2.3.3 Adding a Scattering layer

In machine learning, training accuracy is greatly improved through a process described as feature engineering. In feature engineering, discriminating characteristics of the data is enhanced at the same time non-distinguishing features constituting noise is removed or attenuated to a barest minimum. A lot of the components signal speech signal are due to noise in the environment as well as signal channel distortions such as losses due to conversion from audio signals to electrical signal in the recording system.

In figure 2.2, feature engineering is done at the feature extraction stage of the

ASR pipe line. It has be shown that a common technique using Mel-frequency cepstral coefficients (MFCCs) (Davis and Mermelstein, 1990) can represent speech in a stable fashion that approximate how the working of the human auditory speech processing and is able to filter useful components in the speech signal required for human speech hearing. Similar feature processing schemes have been developed include Perceptual Linear Prediction (PLP) (Hermansky, 1990) and RASTA (Hermansky and Morgan, 1994).

The scattering spectrum defines a locally translation invariant representation of a signal resistant to signal deformation over extended periods of time spanning seconds of the signal (Andén and Mallat, 2014). While Mel-frequency cepstral coefficients (MFCCs) are cosine transforms of Mel-frequency spectral coefficients (MFSCs), the scattering operator consists of a composite wavelet and modulus operation on input signals.

Over a fixed time, MFSCs measure signal energy having constant Q bandwidth Mel-frequency intervals. This procedure is susceptible to time-warping signal distortions since these information often reside in the high frequency regions discarded by Mel-frequency intervals. As time-warping distortions isn't explicit classifier objective when developing these filters, there is no way to recover such information using current techniques.

In addition, short time windows of about 20 ms are used in these feature extraction techniques since at this resolution speech signal is mostly locally stationary. Again, this resolution adds to the loss of dynamic speech discriminating information on signal structures that are non-stationary at this time interval. To minimize this loss Delta-MFCC and Delta-Delta-MFCCs (Furui, 1986) are some of the means developed to capture dynamic audio signal characterisation over larger time scales.

By computing multi-scale co-occurrence coefficients from a wavelet-modulus operation, Andén and Mallat (2011) shows that non-stationary behavior lost by MFSC coefficients is captured by the scattering transform multi scale co-occurrence coefficients and the scattering representation includes MFSC-like measurements. Together with higher-order co-occurrence coefficients, deep scattering spectrum coefficients represents audio signals similar to models based on cascades of constant-Q filter banks and rectifiers. In particular, second-order co-occurrence coefficients carry

important signal information capable of discriminating dynamic information lost to the MFCC analog over several seconds and therefore a more efficient discriminant than the MFCC representation. Second-order co-occurrence coefficients calculated by cascading wavelet filter banks and rectified using modulus operators have been evaluated as equivalent to a light-weight convolutional neural networks whose output posteriors are computed at each layer instead of only at the output layer Mallat (2016).

The premise for this work is that low speech recognition can be achieved by having higher resolution features for discrimination as well as using an end-to-end framework to replace some of the cumbersome and time-consuming hand-engineered domain knowledge required in the standard ASR pipeline. In addition, this research work makes contributions to the requirements for the two tracks specified in the Zero Resource challenge of 2015 (Versteegh et al., 2015). The first requirement is sub-word modelling satisfied with using deep scattering network and the second that of spoken term discovery criteria being satisfied with the end-to-end speech model supplemented with a language model.

2.4 Methodology

System building methodology (Nunamaker Jr et al., 1990) for speech recognition systems require models to be evaluated against speech recognition machine learning metrics. For language models, perplexity metric was used for evaluation. Bleu has also been used as a metric for evaluating language models.

Perplexity measures the complexity of a language that the language model is designed to represent (Jelinek, 1976). In practice, the entropy of a language with an N-gram language model $P_N(W)$ is measured from a set of sentences and is defined as

$$H = \sum_{\mathbf{W} \in \Omega} P_N(\mathbf{W}) \tag{2.9}$$

where Ω is a set of sentences of the language. The perplexity, which is interpreted as the average word-branching factor, is defined as

$$B = 2^H \tag{2.10}$$

$$PP(W) = P(w_1, w_2 \dots w_N)^{\frac{1}{N}} \tag{2.11}$$

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}} \tag{2.12}$$

Full speech recognition pipelines are usually evaluated against the Word Error Rate (WER). WER is computed as follows:

$$WER = \frac{I + D + R}{WC} \times 100 \tag{2.13}$$

Here I, D, and R are wrong insertions, deletions and replacements respectively and WC is the word count.

Metrics used for low speech recognition in the zero speech challenge (Versteegh et al., 2015) includes the ABX metric. Other common speech recognition error metrics following a similar definition as the Word Error Rate (WER) are Character Error Rate (CER), Phoneme Error Rate (PER) and Syllabic Error Rate (SyER) and sentence error rate (SER).

PC Woodland and Daniel Povey. Large scale discriminative training for speech recognition. In *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*, 2000.

Ping Xu and Pascale Fung. Cross-lingual language modeling for low-resource speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 21 (6):1134–1144, 2013.

Steve Young. A review of large-vocabulary continuous-speech. *IEEE Signal Processing Magazine*, 13(5):45, 1996. doi: 10.1109/79.536824.

Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, et al. The htk book. *Cambridge university engineering department*, 3:175, 2002.

Chapter 3

Recurrent Neural Networks in Speech Recognition

The HMM model described in the chapter 2 is uses the divide and conquer strategy which has been defined as a generative method in which we use the smaller components represented by the HMM to learn the entire speech process. As also previously mentioned this can also be referred to as the bottom-up strategy. The discriminative method however uses the opposite mechanism. Rather than using the building blocks of speech to determine speech parameters of a HMM, the discriminative strategy rather determines the posterior probability directly using the joint probability distribution of the parameters involved in the discriminative process. The discriminative parameters are discussed in this section where the Neural network discriminative approach is described beginning with the architecture.

3.1 Neural network architecture

The building block of a neural network simulates a combination of two consecutive linear and non-linear operations having many inputs interconnected with the linear portion of the network. This rudimentary structure is described by McCullough and Pitts (1942) Cowan (1990) as the Perceptron in figure 3.1

The linear operation is the sum of the products between the input features and weight vector. This vector sum of products is referred to as an affine transformation or operation. The non linear operation is the given by any one of a selection of

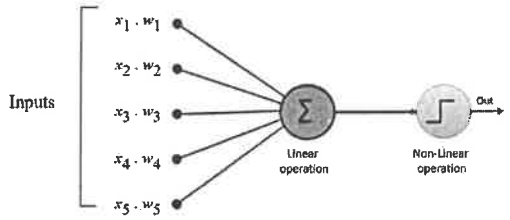


Figure 3.1: Perceptron

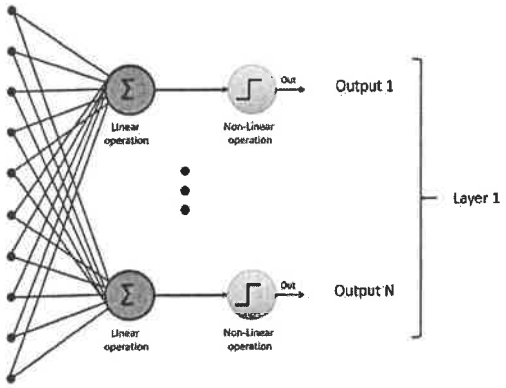


Figure 3.2: Perceptron

nonlinear functions. In figure 3.2 this is shown as a step function. The step function is activated (becomes 1) whenever the output of the linear function is above a certain threshold, otherwise remains at 0. A simple neural network of perceptrons is formed by stacking the perceptrons into an interconnected layer as shown in the figure 3.2 :

In this regime each combination of linear operation followed by a non linear operation is called a neuron and the total number of neurons in the layer formed is termed as N-number of neurons in the layer.

3.1.1 Multi-layer Perceptron (MLP)

The multilayer Perceptron or MLP extends the basic Perceptron structure by adding one or more hidden layers. These hidden layers comprise the outputs of one layer becoming the input of the next layer. In the simplest case having one hidden layer, the output of layer 1 becomes the input of the final output layer. In comparison, the Perceptron is a one dimensional structure having one or more linear and non linear combination outputs, while the multilayer Perceptron is a 2-dimensional structure having one or more hidden layers of N linear and non-linear combination outputs. Mathematically speaking the output of each layer of an MLP having N inputs and

M neurons is given by

$$z_j = h(b_j) = \frac{1}{1 + e^{-b_j}} \tag{3.1}$$

is the non-linear function while is the linear function given by:

$$b_j = \sum_{i=0}^{N} w_{ji}^{(1)} \qquad j = 1, 2, \ldots, M \tag{3.2}$$

For each layer in the MLP, the zeroth input value x_0 is 1 indicating a bias term. This bias term is used in the neural network to ensure regularised and expected behaviour of the neural network. In this example the non-linear step function is given by a more complex exponential. In the next section the nonlinear functions for a multilayer Perceptron is derived.

3.1.2 Sigmoid and soft-max Activation Function

The combination of the linear function and the non linear function in the neural network could be said to be transformation of an algebraic problem to a probabilistic function. In this case the "step" function is a squashing sigmoid-shaped function that converts the inputs into a Naive Bayes function evaluating the probability that an output belongs to any of the output classes (C_y) given the data ($\mathbf{x}$).

$$p(C_1|\mathbf{x}) = f(a) = f(\mathbf{w}^\top \mathbf{x} + w_0) \tag{3.3}$$

In a two class problem ~~with classes and , then we can express~~ the posterior probability of C_1 using Bayes's theorem

$$p(C_1|\mathbf{x}) = \frac{p(\mathbf{x}|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(\mathbf{x}|C_2)p(C_2)} \tag{3.4}$$

Dividing through by $p(\mathbf{x}|C_1)p(C_1)$ gives us

$$p(C_1|(x) = \frac{1}{1 + \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)}} \tag{3.5}$$

If we define the ratio of the log posterior probabilities as

$$a = \ln \frac{p(\mathbf{x}|C_1)p(C_1)}{p(\mathbf{x}|C_2)p(C_2)} \tag{3.6}$$

If we substitute back into (4) we have:

$$p(C_1|\mathbf{x}) = f(a) = \frac{1}{1+e^{-a}} \tag{3.7}$$

Here $a = \mathbf{w}^\top \mathbf{x} = w_0$. Thus the activation for the non-linear function is driven by the probability of the data to give the output class. The probabilistic function here is called a sigmoid function due to the s-shaped graph that is plotted by the function.

Rather than using the sigmoid function for multi-class classification a similar soft max function is derived by using the log probability of classes. If $a_k = \ln(p(\mathbf{x}|C_k)p(C_k))$ then:

$$y_k = p(C_k|\mathbf{x}) = \frac{e^{a_k}}{\sum_{\ell=1}^{K} e^{a_\ell}} \tag{3.8}$$

$$a_k = \sum_{i=0}^{d} w_{ki} x_i \tag{3.9}$$

Recall that in the generative classification method the problem is divided into sub problems by using the conditional probability, while in the discriminative approach the joint probability is determined by looking at the data directly. This is what $p(C_k|\mathbf{x})$ represents. However, recall that we still need to determine the correct probability distribution represented by the data. This is achieved by determining the values of the weights of the linear operation. In the next section a method known as back propagation is discussed. Back propagation is the training algorithm used to determine the weight vector of all the layers in the neural network. Back propagation is an extension of the Gradient descent algorithm.

3.1.3 Back propagation algorithm (backprop)

In the previous section, the neural network architecture has been described as having N inputs M neurons and L layers. Each layer comprises M neurons of a maximum of N inputs times M neurons interconnections which embodies the inner product of the inputs and unknown set of weights. The output of this inner product is then passed to a logistic squashing function that results output probabilities. The discriminative process is used here to determine the correct combination of weight vectors that accurately describe the training data. For neural networks, the weight

vectors at each layer are determined through propagating the errors back through each preceding and adjusting the weights according to the errors propagated each time a batch of the data is processed. This process of continuously adjusting weights from back propagation continues until all the data is processed and a steady state has been reached. The steady state refers to the fact that the error has reached a steady and acceptable value. This is often referred to in machine learning as convergence (Boden, 2002).

Gradient Descent

The last section ended stating that the back-propagation algorithm is an extension of the gradient descent algorithm. It has also been seen that back propagation works by propagating the error and making adjustments on the weights. In this section, the Gradient Descent algorithm is reviewed and how it is used in back propagation is examined.

The concept behind the Gradient descent algorithm is the fact that a function is optimized when the gradient of the function is equal to 0. Gradient descent algorithm is significant in machine learning applications because a cost function is easily defined for a particular machine learning application that is able to determine the error between the predicted value and the actual value. Then, the parameters of the problem can be adjusted until the derivative of the cost function using gradient descent is zero. Therefore the machine learning algorithm adjusts its parameters until the error is minimised or removed.

A common error function or cost function for neural networks is the sum-of-squares error cost function. This is obtained by summing the difference between the actual value and the machine learning model value over the training set N.

$$E^n = \frac{1}{2}\sum_{k=1}^{K}(y_k^n - t_k^n)^2 \tag{3.10}$$

In a neural network having a weight matrix $\mathbf{W}$ of M neurons times N inputs, the resulting gradient is a vector of partial derivatives of E with respect to each element.

$$\nabla_{\mathbf{w}} E = \left(\frac{\partial E}{\partial w_{10}}, \ldots, \frac{\partial E}{\partial w_{ki}}, \ldots, \frac{\partial E}{\partial w_{Kd}}\right) \tag{3.11}$$

The adjustment on each weight therefore on each iteration is:

$$w_{kj}^{\tau+1} = w_{kj}^{\tau} - \eta \frac{\partial E}{\partial w_{kj}} \tag{3.12}$$

Where τ is the iteration and η is a constant learning rate which is a factor to speed up or slow down the rate rate of learning of the machine learning algorithm which in this case is the neural network.

3.2 RNN, LSTM and GRU Networks

Neural networks have become increasingly popular due to their ability to model non-linear system dynamics. Since their inception, there have been many modifications made to the original design of having linear affine transformations terminated with a nonlinear functions as the means to capture both linear and non-linear features of the target system. In particular, one of such neural network modifications, namely the recurrent neural network, has been shown to overcome the limitation of varying lengths in the inputs and outputs of the classic feed-forward neural network. In addition the RNN is not only able to learn non-linear features of a system but has also been shown to be effective at capturing the patterns in sequential data. This section develops recurrent neural networks (RNNs) from a specialised multi-layer Perceptron (MLP) or the deep neural network (DNN).

3.2.1 Deep Neural Networks (DNNs)

Deep neural networks have been accepted to be networks having multiple layers and capable of hierarchical knowledge representation (Yu and Deng, 2016). This will therefore include multi-layer Perceptrons (MLPs) having more than one hidden layer (Dahl et al., 2012) as well as deep belief networks (DBNs)(Mohamed et al., 2009, Yu et al., 2010) having a similar structure. Therefore, following the MLP architecture, A DNN uses multiple hidden layers and generates distribution function, $p(c|x_t)$ on the output layer when an input vector $\mathbf{x}_t$ is applied. At the first hidden layer, activations are vectors evaluated using

$$\mathbf{h}^{(1)} = \sigma(\mathbf{W}^{(1)T}\mathbf{x}_t + \mathbf{b}^{(1)}) \tag{3.13}$$

The matrix $\mathbf{W}^{(1)}$ and vector $b^{(1)}$ are the weight matrix and bias vector for the layer. The function $\sigma(\cdot)$ is the point-wise non-linear function.

DNNs have arbitrarily many hidden layers. After the first hidden layer, the hidden activations $h^{(i)}$ for the layer i are computed as

$$\mathbf{h}^{(1)} = \sigma(\mathbf{W}^{(1)T}\mathbf{h}^{(i-1)} + \mathbf{b}^{(1)}) \tag{3.14}$$

To obtain a proper distribution over the set of possible characters c the final layer of the network is a soft max output layer of the form,

$$p(c = c_k|x_t) = \frac{exp(-(\mathbf{W}_k^{(s)T}h^{(i-1)} + b_k^{(1)}))}{\sum_j exp(-(\mathbf{W}_k^{(s)T}h^{(i-1)} + b_k^{(1)}))} \tag{3.15}$$

where $W_k^{(s)}$ is the k-th column of the output weight matrix $W^{(s)}$ and $b_k^{(k)}$ is the scalar bias term. We can compute the sub gradient for all parameters of the DNN given a training example and thus utilise gradient-based optimisation techniques. This same formulation is used in DNN-HMM models to predict distribution over senones instead of characters.

3.2.2 Recurrent Neural Networks

The two advantages RNNs have over regular DNNs is firstly to capture varying lengths of outputs to inputs. That is for tasks such as language translation for example, there is no one to one correspondence of number of words in a sentence for example from the source language to the output destination language. At the same time the sentence length appearing at the input and that appearing at the output differ for different sentences. This is the first problem of varying lengths for input and output sequences.

The second issue that RNNs effectively contain as opposed to DNNs is capturing temporal relationships between the input sequences. As was realised for hidden Markov models, it was seen that the HMM modeled not just observation likelihoods but also transition state likelihoods which were latent or hidden variables. By tying the output of previous neuron activations to present neuron activations, a DNN is inherits a cyclic architecture becoming a recurrent neural network (RNN). As a

result, an RNN is able capture previous hidden states and in the process derive memory-like capabilities (Yu and Deng, 2016).

In speech processing, it is observed that a given utterance various temporal dependencies which may not be sufficiently captured by DNN-based systems because DNN systems ignore previous hidden representations and output distributions at each time step t. The DNN derives its output using only the feature inputs x_t. The architecture of RNN to enable better model temporal dependencies present in a speech is given in (Hannun et al., 2014b, Yu and Deng, 2016).

$$h_t^{(j)} = \sigma(\mathbf{W}^{(j)T} h_t^{(i-1)} + \mathbf{W}_k^{(j)T} h_{t-1}^{(j)} + b^{(j)})) \tag{3.16}$$

It can be seen in equation (3.16) above given a selected RNN hidden layer j, a temporally recurrent weight matrix $W^{(f)}$ is computed for output activations $h_{t-1}^{(j)}$ for the hidden activation vector of layer j at time step $t-1$ such that the output contributes to the standard DNN output of $\mathbf{W}^{(j)T} h_t^{(i-1)}$. It can also be seen from equation (3.16) that the temporal recurrent weight matrix computation is a modified version of the standard DNN weight matrix computation and that the overall output is a superposition of the two.

Since computations for a RNN are the same as those described in standard DNN evaluations, it is possible to compute the sub gradient for RNN architecture using the back propagation algorithm. The modified algorithm appropriately called back propagation through time (BPTT) (Boden, 2002, Jaeger, 2002) is derived as follows.

3.2.3 LSTMs and GRUs

A special implementation of the RNN called the Long Short Term Memory (LSTM) has been designed to capture patterns over particularly long sequences of data and thus is an ideal candidate for generating character sequences while preserving syntactic language rules learned from the training data.

The internal structure and working of the LSTM cell is documented by its creators in ?. The ability to recall information over extended sequences results from the internal gated structure which performs a series of element wise multiplications on the inputs and internal state of the LSTM cell at each time step. In addition to

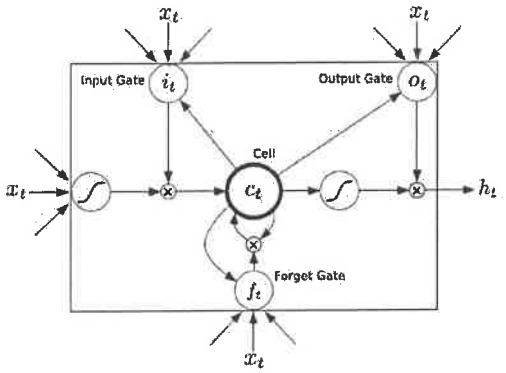


Figure 3.3: An LSTM Cell Graves et al. (2013)

the output neurons which in this text we refer to as the write gate and denote as the current cell state, $\mathbf{c}_t$, three additional gates (comprising a neural network sub-layer) located within the LSTM cell are the input gate, the forget gate and the output gate. Together with the initial current state cell, these gates along with the current-state cell itself enable the LSTM cell architecture to store information, forward information, delete information and receive information. Generally however, the LSTM cell looks like a regular feed-forward network having a set of neurons capped with a nonlinear function. The recurrent nature of the network arises, however due to the fact that the internal state of the RNN cell is rerouted back as an input to the RNN cell or input to the next cell in the time-series give rise to sequence memory within the LSTM architecture. Mathematically, these gates are formulated as follows:

$$\mathbf{i}_t = \sigma(\mathbf{W}^{(xi)}\mathbf{x}_t + \mathbf{W}^{(hi)}\mathbf{h}_{t-1} + \mathbf{W}^{(ci)}\mathbf{c}_{t-1} + \mathbf{b}^{(i)}) \tag{3.17}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}^{(xf)}\mathbf{x}_t + \mathbf{W}^{(hf)}\mathbf{h}_{t-1} + \mathbf{W}^{(cf)}\mathbf{c}_{t-1} + \mathbf{b}^{(f)}) \tag{3.18}$$

$$\mathbf{c}_t = \mathbf{f}_t \bullet \mathbf{c}_{t-1} + \mathbf{i}_t \bullet \tanh(\mathbf{W}^{(xc)}\mathbf{x}_t + \mathbf{W}^{(hc)}\mathbf{h}_{t-1} + \mathbf{b}^{(c)}) \tag{3.19}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}^{(xo)}\mathbf{x}_t + \mathbf{W}^{(ho)}\mathbf{h}_{t-1} + \mathbf{W}^{(co)}\mathbf{c}_{t-1} + \mathbf{b}^{(o)}) \tag{3.20}$$

$$\mathbf{h}_t = \mathbf{o}_t \bullet \tanh(\mathbf{c}_t) \tag{3.21}$$

The gates in the above formula are illustrated in Figure 3.3. $\mathbf{i}_t$ represents the input gate, $\mathbf{f}_t$ is the forget gate and $\mathbf{o}_t$ represents the output gate. At each of these gates therefore, the inputs consisting of hidden states in addition to the regular inputs are multiplied by a set of weights and passed through a soft-max function.

These weights during training learn whether the gate will, during inference, open or not. In summary, the input gate tells the LSTM not whether or not to receive new information, the forget gate determines whether the current information it already has from the previous step should be kept or dropped and the output gate determines what should be forwarded to the next LSTM cell. Note also that the LSTM has two sigmoid (*tanh*) activation functions utilised at the input and output of the current cell $\mathbf{c}_t$.

One particular variant of the original LSTM model is the GRU cell. Though simpler than an LSTM cell the GRU cell performs equally efficient. The main simplifications are that both state vectors are merged into a single vector $\mathbf{h}_{(t)}$. A single gate controller controls both the forget gate an the input gate. If the gate controller outputs a 1, the input gate is open and the forget gate is closed. If it outputs a 0, the opposite happens. In other words, whenever a memory be stored, the location where it will be stored is erased first. This is actually a frequent variant LSTM cell in and of itself. Finally, there is no output gate; the full vector cost is output at every time step. However, there is a new gate controller that controls which part or the previous state will be shown to the main layer.

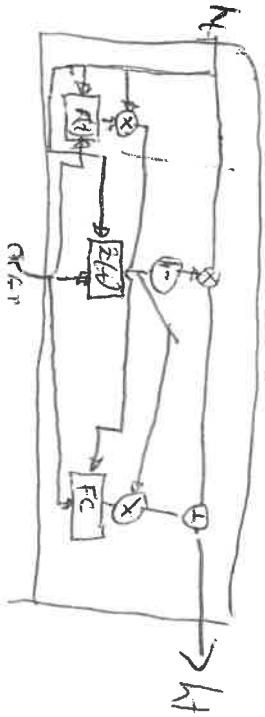
The overall architecture of a GRU is as follows:

$$z_{(t)} = \sigma\left(W_{xz}^{T} \cdot x_{(t)} + W_{hz}^{T} \cdot x_{(t-1)}\right)$$

$$r_{(t)} = \sigma\left(W_{xr}^{T} \cdot x_{(t)} + W_{hr}^{T} \cdot x_{(t-1)}\right)$$

$$g_{(t)} = \tanh\left[W_{xg}^{T} \cdot x_{(t)} + W_{hg}^{T} \cdot \left(r_{(t)} \otimes h_{(t-1)}\right)\right]$$

$$h_{(t)} = \left(1 - z_{(t)}\right) \otimes \left(h_{(t-1)}\right) + z_{(t)} \otimes g_t$$



3.3 Deep speech architecture

This work makes use of an enhanced RNN architecture called the Bi-directional Recurrent Neural Network (BiRNN). While Hannun et al. (2014b) assert that forward recurrent connections does reflect the sequential relationships of an audio waveform, perhaps the BiRNN model poses a more powerful sequence model.

The BiRNN is a preferred end to end mechanism due to the length of sequence over which temporal relationships can be captured. This implies that BiRNNs will be suited for capturing temporal relationships over much longer sequences than a forward only RNN, because hidden state information are preserved in both forwards and backwards direction.

In addition, such a model has a notion of complete sentence or utterance integration, having information over the entire temporal extent of the input features when

making each prediction.

The formulation of the BiRNN is derived by starting of with the basic RNN architecture which is referred to as the forward architecture. From the forward architecture we derive the backward architecture. If we choose a temporally recurrent layer j, the BiRNN forward and backward intermediate hidden representation $h_t^{(f)}$ and $h_t^{(b)}$ is given as.

$$h_t^{(f)} = \sigma(\mathbf{W}^{(j)T} h_t^{(i-1)} + \mathbf{W}_k^{(f)T} h_{t-1}^{(j)} + b^{(j)})) \tag{3.22}$$

$$h_t^{(b)} = \sigma(\mathbf{W}^{(j)T} h_t^{(i-1)} + \mathbf{W}_k^{(b)T} h_{t+1}^{(b)} + b^{(j)})) \tag{3.23}$$

Temporal weight matrices $W^{(f)}$ and $W^{(b)}$ propagate $h_t^{(f)}$ and $h_t^{(b)}$ forward and backward in time respectively.

? points out that the recurrent forward and backward components are evaluated entirely independent of each other and for optimal training, a modified non linearity function $\sigma(z) = min(max(z, 0), 20)$ is recommended.

The final BiRNN representation $h_t^{(j)}$ for the layer is now the sum of the two RNN components,

$$h_t^{(j)} = h_t^{(f)} + h_t^{(b)} \tag{3.24}$$

Also note that ~~back-propagation~~ sub gradient evaluations is computed from the combined BiRNN structure directly during training.

3.3.1 Connectionist Temporal Classification (CTC)

Bibliography

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International Conference on Machine Learning*, pages 173–182, 2016.

Joakim Andén and Stéphane Mallat. Multiscale scattering for audio classification. In *ISMIR*, pages 657–662. Miami, FL, 2011.

Joakim Andén and Stéphane Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, 2014.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in neural information processing systems*, pages 153–160, 2007.

Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014.

Peter F. Brown, Peter V. Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.13.9919rep=rep1type=pdf`

Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve. Wav2letter: an end-to-end convnet-based speech recognition system. *arXiv preprint arXiv:1609.03193*, 2016.

Steven B Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In *Readings in speech recognition*, pages 65–74. Elsevier, 1990.

Li Deng and Xiao Li. Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(5): 1060–1089, 2013.

Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

Sadaoki Furui. Speaker-independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(1):52–59, 1986.

Arnab Ghoshal, Pawel Swietojanski, and Steve Renals. Multilingual training of deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7319–7323. IEEE, 2013.

Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772, 2014.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 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, 2006.

Frantisek Grezl and Petr Fousek. Optimizing bottle-neck features for lvcsr. In *ICASSP*, volume 8, pages 4729–4732, 2008.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014a.

Awni Y Hannun, Andrew L Maas, Daniel Jurafsky, and Andrew Y Ng. First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. *arXiv preprint arXiv:1408.2873*, 2014b.

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

Hynek Hermansky and Nelson Morgan. Rasta processing of speech. *IEEE transactions on speech and audio processing*, 2(4):578–589, 1994.

F. Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, 1976. doi: 10.1109/PROC.1976.10159.

Bing-Hwang Juang and S. Furui. Automatic recognition and understanding of spoken language - a first step toward natural human-machine communication. *Proceedings of the IEEE*, 88(8):1142–1165, 2000. doi: 10.1109/5.880077. URL `http://ieeexplore.ieee.org/document/880077`.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *AAAI*, pages 2741–2749, 2016.

Julius Kunze, Louis Kirsch, Ilia Kurenkov, Andreas Krug, Jens Johannsmeier, and Sebastian Stober. Transfer learning for speech recognition on a budget. *arXiv preprint arXiv:1706.00290*, 2017.

Thang Luong, Richard Socher, and Christopher D Manning. Better word representations with recursive neural networks for morphology. In *CoNLL*, pages 104–113, 2013.

Stéphane Mallat. Understanding deep convolutional networks. *Phil. Trans. R. Soc. A*, 374(2065):20150203, 2016.

Tomáš Mikolov, Anoop Deoras, Stefan Kombrink, Lukáš Burget, and Jan Černockỳ. Empirical evaluation and combination of advanced language modeling techniques. In *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

Abdel-rahman Mohamed, George E Dahl, Geoffrey Hinton, et al. Acoustic modeling using deep belief networks. *IEEE Trans. Audio, Speech & Language Processing*, 20(1):14–22, 2012.

Jay F Nunamaker Jr, Minder Chen, and Titus DM Purdin. Systems development in information systems research. *Journal of management information systems*, 7 (3):89–106, 1990.

Daniel Povey, Lukáš Burget, Mohit Agarwal, Pinar Akyazi, Feng Kai, Arnab Ghoshal, Ondřej Glembek, Nagendra Goel, Martin Karafiát, Ariya Rastrow, et al. The subspace gaussian mixture model—a structured model for speech recognition. *Computer Speech & Language*, 25(2):404–439, 2011.

Charles Ogan D. S. *Okrika: A kingdom of the Niger Delta.* Onyoma Research Publications, Port Harcourt, Rivers State, Nigeria, 1 edition, 2008.

George Saon, Hong-Kwang J Kuo, Steven Rennie, and Michael Picheny. The ibm 2015 english conversational telephone speech recognition system. *arXiv preprint arXiv:1505.05899*, 2015.

Gary F. Simons and Charles D. Fennig. Ethnologue: Languages of the world, twenty-first edition., 2018. URL `http://www.ethnologue.com`.

Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. Technical report, COLORADO UNIV AT BOULDER DEPT OF COMPUTER SCIENCE, 1986.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Maarten Versteegh, Roland Thiolliere, Thomas Schatz, Xuan Nga Cao, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. The zero resource speech challenge 2015. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

Ngoc Thang Vu and Tanja Schultz. Multilingual multilayer perceptron for rapid language adaptation between and across language families. In *Interspeech*, pages 515–519, 2013.

Shinji (. e. Watanabe and Jen-Tzung Chien. *Bayesian speech and language processing.* Cambridge University Press, Cambridge, 2015. ISBN 1107055571;9781107055575;.