Chapter 5

Wakirike Language Model

A language model for the Wakirike language is developed in this chapter. This model draws upon the premise that the grammar of a language is expressed in the character sequence pattern which is ultimately expressed in words and therefore the abstract grammar rules can be extracted and learned by a character-based RNN neural network.

5.1 Data Preparation

The Wakirike New Testament Bible served as the source of data for the deep neural network training. As there wasn't a readily available soft or on-line copy of the Wakirike new testament bible for use, this was typed, giving rise to a complete corpus word size of 668,522 words and a character count of 6,539,176 characters. The data set was then divided into 11 parts. Two parts dedicated for testing and validation and the remaining 9 parts were used for training.

The Unicode representations of the character set consisting of letters and punctuation marks are one-hot encoded and batched for sequential input, each batch having a character sequence length of 30 characters.

5.2 GRU Training

The modified LSTM RNN known as the Gated Recurrent Unit (GRU) is employed for the neural network model in order to optimise network performance in terms of resource conservation. GRUs have been shown to give similar performance to regular LSTMs however, with a lighter system resource footprint (Cho et al., 2014). The GRU RNN used to train the Wakirike text corpus comprised an internal network size of 512 nodes for each layer and was 3 layers deep. Externally, 30 GRUs represented the number of recurrent connections each connection representing a time step bearing contextual for the recurrent input sequence.

To mitigate for over-fitting, due to the multi-layered high-dimensional depth of this neural network, a small learning rate of 0.001 was used. To further marginalise over-fitting the popular and effective dropout (Srivastava et al., 2014) method for regularising deep neural networks kept at 20% such that only 80% activations were propagated from one layer to the next and the remaining 20% were zeroed out.

5.3 Output Language Generation

The neural network is trained for 10 epochs and achieves a prediction accuracy of 85% on held-out data. With the GRU model it is possible to seed this network with an input character and select from the top-N candidates thus causing the Neural network to generate its own sentences. In this scenario, the network is said to perform language generation by immanently constructing its own sentences. The generated language was found to be intelligibly similar to that of the training data.

A clever use of this new corpus generated by the GRU language model of this work was to determine a word-based perplexity metric for the GRU neural language model. In this work, the word-based perplexity metric was achieved from the output language generated by first estimating the word based perplexity on the training data. The same perplexity calculation was then used on the generated neural language model corpus. The corpus size of the neural language model was made to be equivalent to that of the training data, that is containing 6,539,176 characters. The perplexity calculation was based on a modified Kneser-Key 5-gram model with smoothing (Heafield et al., 2013). The results discussed below showed that the LSTM model generated a superior model compared to the n-gram model that better matched the training data.

The evaluation of the GRU language model of the Wakirike language was performed using a perplexity measurement metric. The Perplexity metric applies the language model to a test dataset and measures how probable the test dataset is. Perplexity is a relative measure given by the formula:

$$PP(W) = P(w_1, w_2 \dots w_N)^{\frac{1}{N}} \tag{5.1}$$

$$PP(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_{i-1})}} \tag{5.2}$$

Where $w_1, \dots, w_N$ are the sequence of words. The language model with the lower relative perplexity score is therefore expected to yield better approximation of the data when applied to unseen data generally. The result of the training of the GRU-Cell Recurrent Neural Network on low-resourced Wakirike Language gave impressive and intelligible results and showed better results when measured with standard n-gram language models. The results showed that it is indeed possible to use an LSTM on a low resource character sequence corpus to produce an Wakirike language model.

A character based perplexity metric is possible using the negative log likelihood of the character sequence.

$$PP(X) = exp\left\{\frac{\sum_{t=1}^{T} \log P(x_t|x_{1:t1})}{T}\right\} \tag{5.3}$$

However, our base-line language model is a 5-gram word-based language model. Therefore, comparing a word based model to a character based model requires a conversion step. In this work, the conversion step involved using the GRU language model generated a corpus which was rescored by re-estimating with a 5-gram word-based language model

Table 5.1 shows the Results of the Perplexity model of the LSTM Wakirike Language model and an equivalent 5-gram Language model with interpolation and Keysner smoothing (Heafield et al., 2013).

5.4 Summary

It can be inferred that the GRU character-model developed has an improved language model and because it is based on a character-model, which is fine-grained

Table 5.1: Perplexity Calculation results

Language Model	Perplexity	
Held-out data size (characters)	998	99
LSTM RNN	1.6398	1.7622
3-gram with Keysner Soothing and interpolation	1.8046	1.9461

when compared to a word model, it is likely to generalise data better when used in practice is and less biased than a word-based model. This can be observed from the fact that the output corpus produced a larger vocabulary size.

Chapter 6

Deep Learning Speech Model

This work explores the prospects of deep recurrent end-to-end architectures applied to speech recognition. Complementary aspects of developing speech recognition systems are eliminated by focusing on end-to-end speech units as a two-step process requiring a Connectionist Temporal Character Classification (CTC)Graves et al. (2006) model and Language Model (LM) rather than a three-step process requiring an Acoustic model(AM), LM and phonetic dictionary. A two-step process rather than a three-step process is particularly desirable for low resource languages as less effort is required developing fewer simplified models.

Earlier in chapter one, deep learning was defined as a type of representational learning whereby different levels of complexity are captured in internal layer-wise encapsulations. It has also been noted that layer-wise stacking of neural and neural network type architectures such as the Restricted Boltzmann Machine (RBM) deep belief networks (DBMs) are used to implement such representations. In this chapter, the end-to-end Bi-directional Recurrent Neural Network model is described. Here, the development of the features using the deep scattering convolution network is first elaborated on. The model parameters and architecture is described and the decoding algorithm is detailed.

6.1 Deep Scattering Features

The fast wavelet transform is derived in Chapter 4.5 from a low pass filter and a high pass filter. The speech features used in this research using a deep scattering network 2 layers deep was created using the wavelet modulus operator comprising a low pass filter and a band pass filter. Hyper parameters of the system included the window period for each sampled sub section, T; The Q-band value for the band pass filter and the number of wavelets J at each scattering layer for the total number of layers, $M = 2$.

The matlab scatnet toolbox (Andén et al., 2014), used to determine the scatter coefficient features for this research, provides optimal values for hyper parameters for audio signal processing into scatter features. In this regime the value for the hyper parameter $T = 512$ samples per window. This corresponds to a window of $50 milliseconds$ for the audio signals sampled at $8000Hz$. For the first scattering layer the Q-band parameter was $Q = 8$ and the second scattering layer took the value $Q = 1$. Finally J is pre-calculated based on the value of T. These after Scat-Net processing produce a feature-vector having 165 dimensions. These feature vectors in turn are used as inputs to the bi-direction neural network model whose architecture is described in succeeding sections.

6.2 CTC-BiRNN Architecture

The core of the system is a bidirectional recurrent neural network (BiRNN) trained to ingest scatter coefficients described in the previous section, in order to generate English text transcriptions. An end-to-end system therefore specifies that utterances x and the corresponding label y be sampled from a training set such that the sample $S = (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ...$ In our end-to-end model, each utterance, $x^{(i)}$ is a processed feature vector consisting of 165 dimensions. Recall, every window passes through a scattering transform to yield an input of vector of $p = 165$ features; consequently, $x^{(i)}_{t,p}$ denotes the p-th feature in a scatter transform at time t.

GPU training of the speech model architecture developed above was done using Mozilla deepspeech (moz, 2019a) CTC bi-directional RNN implementation along with the accompanying Mozilla Common voice dataset (moz, 2019b). The Common Voice Dataset project consists of voice samples in short recordings approximately 4 seconds each. The complete dataset is about 250 hours of recording divided into training, test and development subsets. The BiRNN, given the input sequence, x, outputs a sequence of probabilities $y_t = P(c_t|x)$, where $c_t \in a, b, c, \ldots, z, space, apostrophe, blank$.

The actual architecture of our core Bi-RNN is similar to the deepspeech system described in Hannun et al. (2014a). This structure constitutes 5 hidden layers and one output layer. The first three layers are regular DNNs followed by a bi-directional recurrent layer. As such, the output of the first three layers are computed by:

$$h_t^{(l)} = g(W^{(l)} h_t^{(l1)} + b^{(l)}) \tag{6.1}$$

$g(\cdot) = min\{max\{0, z\}, 20\}$ is the clipped rectified linear unit and $W^{(l)}, b^{(l)}$ are weight matrix and bias parameters for layer as described in sections 3.2.1 and 3.3 respectively.

It was shown in chapter 3 the recurrent layer comprise a forward and backward RNNs whose equations are repeated here for reference

$$h_t^{(f)} = g(W^{(4)} h_t^{(3)} + W_r^{(f)} h_{t1}^{(f)} + b^{(4)}) \tag{6.2}$$

$$h_t^{(b)} = g(W^{(4)} h_t^{(3)} + W_r^{(b)} h_{t+1}^{(b)} + b^{(4)}) \tag{6.3}$$

Consequently, $h^{(f)}$ is the sequential computation from $t = 1$ to $t = T^{(i)}$ for the i-th utterance and $h^{(b)}$ is the reverse computation from $t = T^{(i)}$ to $t = 1$. In addition the output from layer five is summarily given as the combined outputs from the recurrent layer:

$$h^{(5)} = g(W^{(5)} h^{(4)} + b^{(5)}) \tag{6.4}$$

where $h^{(4)} = h^{(f)} + h^{(b)}$. The output of the Bi-RNN on layer 6 is a standard soft-max layer that outputs a predicted character over probabilities for each time slice t and character k in the alphabet:

$$h_{t,k}^{(6)} = \hat{y}_{t,k} \equiv P(c_t = k \mid x) = \frac{\exp\left((W^{(6)} h_t^{(5)})_k + b_k^{(6)}\right)}{\sum_j \exp\left((W^{(6)} h_t^{(5)})_j + b_j^{(6)}\right)}) \tag{6.5}$$

$b_k^{(6)}$ takes on the -th bias and $(W^{(6)} h_t^{(5)})_k$ is the matrix product of the k-th element. The error of the outputs are then computed using the CTC loss function Graves (2014) as described in chapter 3. A summary of our model is illustrated in

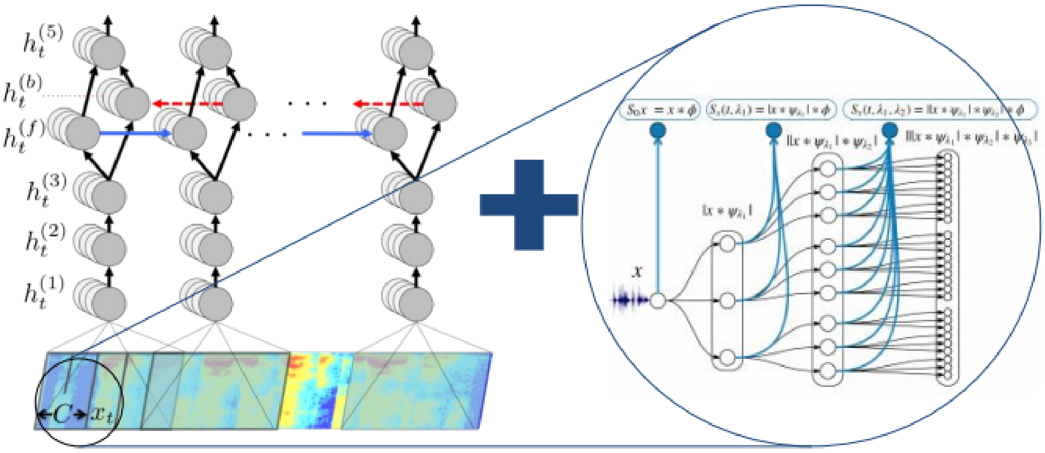
$h_t^{(5)}$
$h_t^{(b)}$
$h_t^{(f)}$
$h_t^{(3)}$
$h_t^{(2)}$
$h_t^{(1)}$
C
x_t
$S_0x = x * \phi$
$S_1(t, \lambda_1) = |x * \psi_{\lambda_1}| * \phi$
$S_1(t, \lambda_1, \lambda_2) = ||x * \psi_{\lambda_1}| * \psi_{\lambda_2}| * \phi$
$||x * \psi_{\lambda_1}| * \psi_{\lambda_2}|$
$|||x * \psi_{\lambda_1}| * \psi_{\lambda_2}| * \psi_{\lambda_3}|$
$|x * \psi_{\lambda_1}|$
x

Figure 6.1: Deep scattering Bi-RNN Model

figure 6.1.

6.3 CTC Decoding

In chapter three the CTC loss function algorithm was established as being able to maximise the probability of two cases. The first case of transiting to a blank and the second case of transiting to a non blank. In this section, this concept is used to enable decoding of the network output from posterior distribution output to character sequences which can be measured against a reference transcription using either character error rate (CER) or word error rate (WER).

Recall, all the symbols in the alphabet Σ and augmented with the blank symbol. The posterior output of the CTC network is probability of the symbol given the speech feature input $p(c|x_t)$ at time t for $t = 1, \ldots, T$ and T is the length of the input sequence. Also recall two further sets of probabilities also being maintained by the model being the probability of a blank character p_b and that of a non blank character p_{nb}.

Several strategies have been employed to obtain a translation string from the output of the deep neural network. The prefix beam search employed by the CTC decoder of this research is derived from an initial greedy approximation, where at each time step determine the argument that maximises the probability $p(c|x_t)$ at each time step. Let $C = (c_1, \ldots, c_T$ be the character string then, the greedy approach has

$$c_t = arg \max_{c \in \Sigma} p(c|x_t) \tag{6.6}$$

However, this simple approximation is unable to collapse repeating sequences and remove blank symbols. In addition, the approximation is unable to include the constraint of a lexicon or language model.

Prefix beam search algorithm Hannun et al. (2014b) adopted in this work incorporates a language model derived from a lexicon in addition to keeping track of the various likelihoods used for decoding. For the language model constraint, the transcription W is recovered from acoustic input X at time t by choosing the word which maximising the posterior probability:

$$W_i = arg \max_{W_i \in \Sigma_W} p_{net}(W; X) p_{lm}(W) \tag{6.7}$$

In equation 6.7, the Bayes product of language model prior p_{lm} and the network output p_{net} are utilised to maximise the probability of a particular character-word sequence in the lexicon given by Σ_W. The overall calculation used to derive the final posterior distribution includes word insertion factors (α and β) used to balance the highly constrained n-gram language model.

The second strategy adopted by the prefix beam search which improves the decoding algorithm is the beam search strategy. With this approach, the search maintains all possible paths however retains only k number paths which maximise the output sequence probability. Improvements gained with this method are seen when certain maximal paths are made obsolete owing to new information derived from the multiple paths in being maintained in memory.

The recursive prefix beam search algorithm illustrated in figure 6.2 attempts to find the string formulated in equation 6.7. Two sets prefixes A_{prev} and A_{nxet} are initialised, such that at A_{nxet} maintains the prefixes in the current time-step while A_{prev} maintains only k-prefixes from the previous time-step. Note that at the end of each time step A_{prev} is updated with only -most probable prefixes from A_{nxet}. Therefore while, A_{nxet} contains all the possible new paths from based on A_{prev} as a Cartesian product of $A_{prev} \times \Sigma \in \mathcal{Z}^k \times \mathcal{Z}^{|\Sigma|}$ where $|\Sigma|$ is the length of Σ. The probabilities of each prefix obtained at each time step is the sum of the probability of non-blank plus the probability of a blank symbol.

At every time step and for every prefix ℓ currently in A_{prev}, a character from the alphabet Σ is presented to the prefix. The prefix is only extended only when the presented symbol is not a blank or a space. $A_{nxet}$and A_{prev} maintain a list of active prefixes at the previous time step and proposed prefixes at the next time step respectively, The prefix probability is given by multiplying the word insertion term by the sum of the blank and non-blank symbol probabilities.

$$p(\ell|x_{1:t}) = (p_{nb}(\ell|x_{1:t}) + p_b(\ell|x_{1:t}))|W(\ell)|^{\beta} \tag{6.8}$$

$W(\cdot)$ is obtained by segmenting all the characters in the sequence with the space-character symbol and truncating any characters trailing the set of words in the sequence. The prefix distribution however varies slightly depending on network output character being presented.

ℓ_{end} is the variable representing the last symbol in the prefix sequence in A_{prev}. If the symbol presented is the same as ℓ_{end} then the probability of a non-blank symbol,$p_{nb} = 0$. If the symbol being presented is blank then we do not extend the prefix. Finally, if the symbol being presented is a space then we invoke the language model as follows

$$p(\ell^{+}|x_{1:t}) = p(W(\ell^{+})|W(\ell))^{\alpha}(p_{nb}(\ell|x_{1:t}) + p_b(\ell|x_{1:t}))|W(\ell)|^{\beta} \tag{6.9}$$

Note that $p(W(\ell^{+})|W(\ell))$ is set to 0 if the current word $W(\ell^{+})$ is not in the lexicon. This becomes a constraint to enforce all character strings to constitute of words only in the lexicon. Furthermore, $p(W(\ell^{+})|W(\ell))$ is extended to include all the character sequences representing number of words considered by the n-gram language model by constituting the last $n-1$ words in character sequence $W(\ell)$.

6.4 Model Hyper parameters

The hidden layer matrix for each layer comprised 1024 hidden units (6.6M free parameters). The weights are initialised from a uniform random distribution having a standard deviation of 0.046875. Adam optimisation algorithm (Kingma and Ba,

Table 6.1: GPU Experiments

Experiment	Hours of speech	Total training time	Estimated training
1. 2xGPU 10GB RAM	1	7 days	Completed
2. 2xGPU 10GB RAM	10	150+ days	300 days
3. 5xGPU 15GB RAM	10	17 hours	Completed
4. 5xGPU 15GB RAM	40	5+ days	10 days

2014) was used with initial learning rate of , and a momentum of 0.95 was deployed to optimise the learning rate.

The network for network was trained for a total of five to fifty epochs over the training set for experiments conducted. The training time for Python GPU implementation is shown in Table 6.1. For decoding with prefix search we use a beam size of 200 and cross-validated with a held-out set to find optimal settings for the parameters α and β. Figure 6.4 shows word error rates for various GPU configurations and audio data-set sizes.

6.5 Results

A total of four experiments were carried out on two different GPU configurations. A set of experiments was performed a GPU configuration consisting of 2 GPUs having a total of 10 gigabytes of memory. The second set of experiments was carried out on a GPU configuration comprising 5 GPUs having a total of 15 gigabytes of memory. For each configuration two experiments were carried out on a small subset of the dataset then on a larger subset of the common voice dataset being used. The various GPU configurations along with the training times is shown in Table 1.

The output of the training produced mostly gibberish when trained in both configurations using only just one hour of training data. Training loss reduced significantly once the data was increased to ten hours of training. However word error rates (WER) only showed improvement on the 40 hours dataset.

The results showed that the training of the model was heading towards a very slow convergence as indicated by the slow decrements in training loss. However, we perceive that given the complete dataset to train the model will not only converge but show improvements in word error rates.

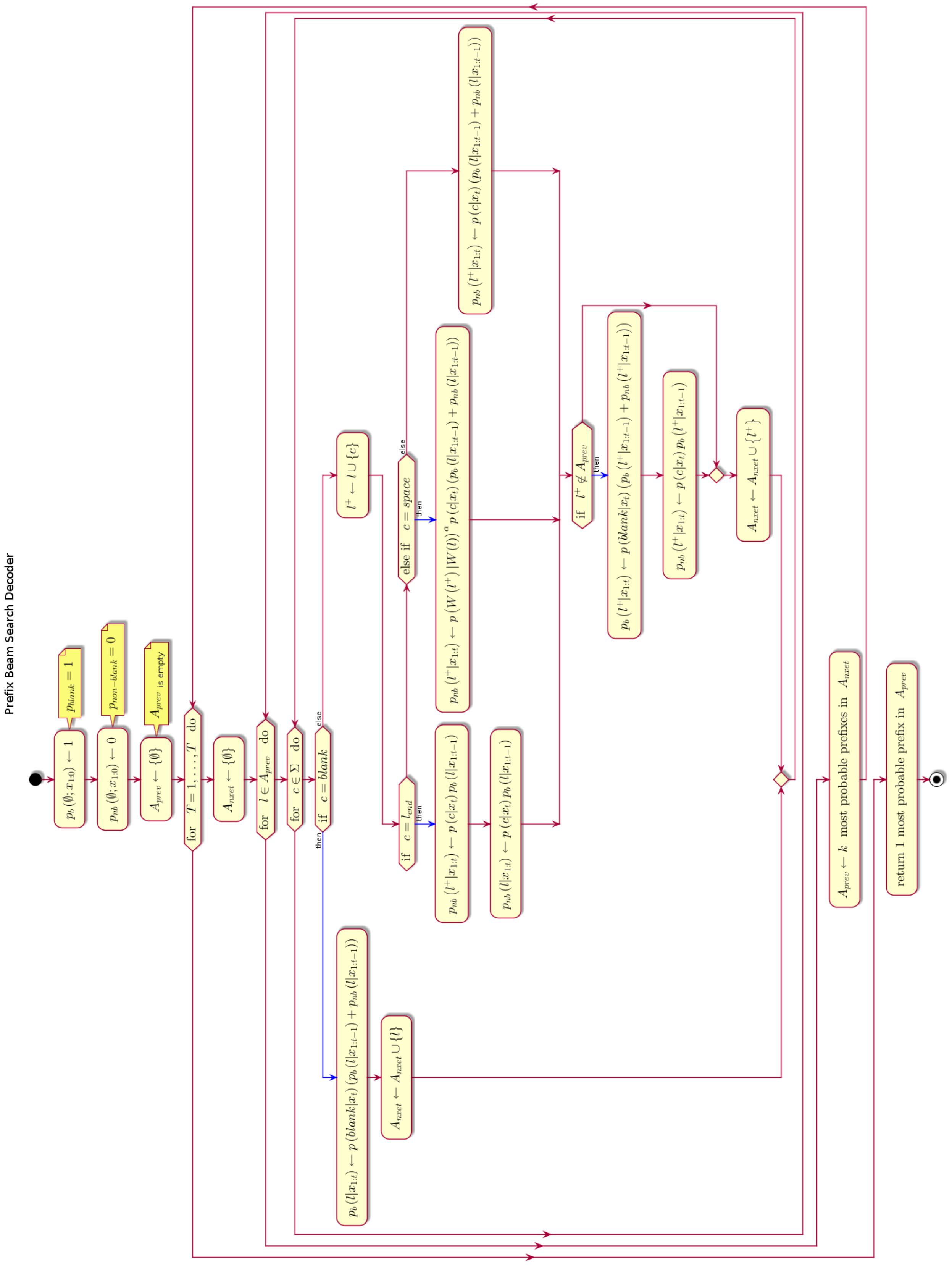
Prefix Beam Search Decoder
$p_b(\emptyset; x_{1:0}) \leftarrow 1$
$p_{blank} = 1$
$p_{nb}(\emptyset; x_{1:0}) \leftarrow 0$
$p_{non-blank} = 0$
$A_{prev} \leftarrow \{\emptyset\}$
A_{prev} is empty
for $T = 1, \ldots, T$ do
$A_{next} \leftarrow \{\emptyset\}$
for $l \in A_{prev}$ do
for $c \in \Sigma$ do
if $c = blank$
then
$p_b(l|x_{1:t}) \leftarrow p(blank|x_t)(p_b(l|x_{1:t-1}) + p_{nb}(l|x_{1:t-1}))$
$A_{next} \leftarrow A_{next} \cup \{l\}$
else
$l^+ \leftarrow l \cup \{c\}$
if $c = l_{end}$
then
$p_{nb}(l^+|x_{1:t}) \leftarrow p(c|x_t)p_b(l|x_{1:t-1})$
$p_{nb}(l|x_{1:t}) \leftarrow p(c|x_t)p_b(l|x_{1:t-1})$
else if $c = space$
then
$p_{nb}(l^+|x_{1:t}) \leftarrow p(W(l^+)|W(l))^{\alpha} p(c|x_t)(p_b(l|x_{1:t-1}) + p_{nb}(l|x_{1:t-1}))$
else
$p_{nb}(l^+|x_{1:t}) \leftarrow p(c|x_t)(p_b(l|x_{1:t-1}) + p_{nb}(l|x_{1:t-1}))$
if $l^+ \notin A_{prev}$
then
$p_b(l^+|x_{1:t}) \leftarrow p(blank|x_t)(p_b(l^+|x_{1:t-1}) + p_{nb}(l^+|x_{1:t-1}))$
$p_{nb}(l^+|x_{1:t}) \leftarrow p(c|x_t)p_b(l^+|x_{1:t-1})$
$A_{next} \leftarrow A_{next} \cup \{l^+\}$
$A_{prev} \leftarrow k$ most probable prefixes in A_{next}
return 1 most probable prefix in A_{prev}

Figure 6.2: Prefix beam search algorithm

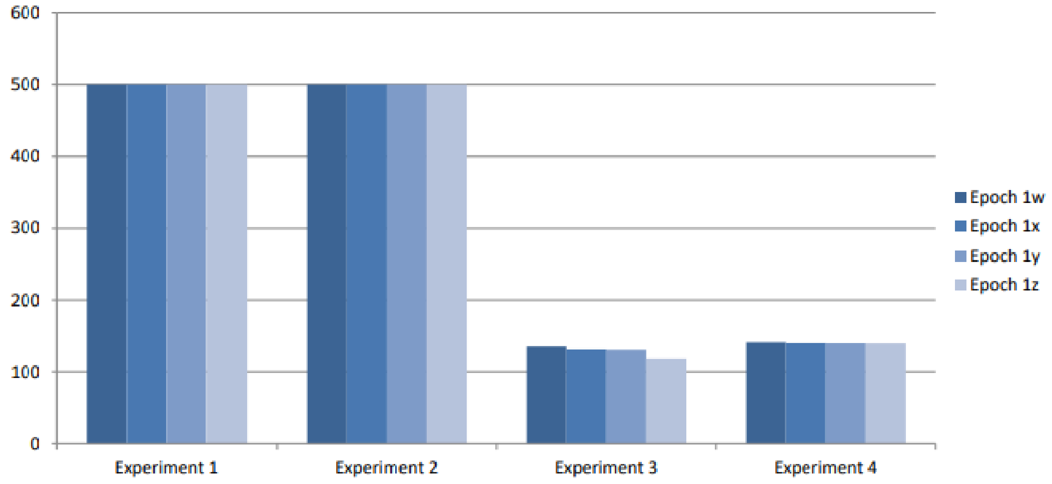

Figure 6.3: Training Loss, where $w < x < y < z$ are taken arbitrarily across the total number of epochs

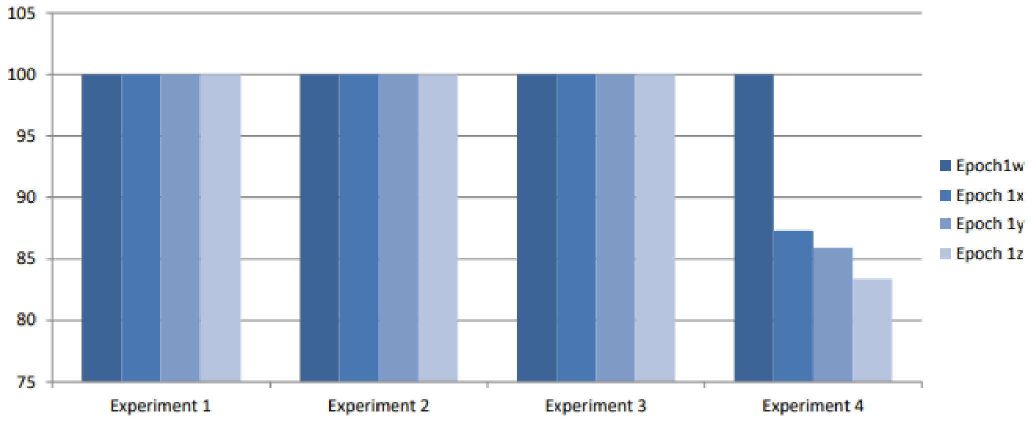

Figure 6.4: WER, where w¡x¡y¡z are taken arbitrarily across the total number of epochs

Chapter 7

Future study

The advancement of machine learning has a direct impact on the development of more efficient speech recognition algorithms and at the same time the advancement of speech recognition helps with the improvement of machine learning algorithms, as in general, the methods used in machine learning usually are directly transferable to speech processing and vice-versa. This mutual relationship implies that speech recognition is a blossoming research field because there is a tremendous amount of work being done in the machine learning community. Particularly in the area of deep learning and neural networks, there is quite a vast array of neural network solutions that have been applied or are yet to be applied to speech recognition. Two models worthy of mentioning are Generative Adversarial Networks (GANs) and Attention-based models.

7.1 Generative adversarial networks (GAN)

GANs consists of two Networks working as adversaries to one another. The first being a generative network generates content. The second network is a discriminative network to determine the accuracy of the first generative network. Hence the generative network is generating output less distinguishable for the discriminator while the discriminator uses output from the generator to improve it's ability to discriminate output from the generator with the original data.

GAN networks can have application where the generative network consists of a speech synthesis network and the discriminating network is a speech recogniser.

However successive training of these two networks from a data-resource perspective would require an immense amount of data resources for expected performances.

7.2 Attention-based Models

The objective of attention-based networks highlighted by Vaswani et al. (2017) is to reduce sequential computation while attaining hidden representation across arbitrary lengths of sequential input. Mechanisms which have been deployed to achieve this includes a combination of convolutional and recurrent schemes Gehring et al. (2017), Kaiser and Bengio (2016), Kalchbrenner et al. (2016). Vaswani et al. (2017) introduces a transduction model known as a Transformer based on self attention network with the ability to compute long term dependencies while eliminating sequence aligned RNN and convolutional architectures.

Self attention is a network that intrinsically reduces the need for intensive resource training. Vaswani et al. (2017) reports that state of the art BLEU score of 41.0 having used a small fraction of training resources. While GANs might not be attractive for low resource speech recognition. They still remain an important tool for verification of the output of other networks at the same time self attention networks can help to the resource needs of GANs when applied to a GAN.

As a further to this thesis, these networks are likely candidates for network training using scatter features as input discriminatory functions. Attention based networks as a means reduce training resources required while GANs can be used as a means to generate training data.

Bibliography

2019a. URL `https://github.com/mozilla/DeepSpeechcommon-voice-training-data`.

2019b. URL `https://voice.mozilla.org/en`.

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.

J Andén, L Sifre, S Mallat, M Kapoko, V Lostanlen, and E Oyallon. Scatnet (v0. 2). *Computer Software. Available: http://www.di.ens.fr/data/software/scatnet/. [Accessed: December 10, 2013]*, 0.2, 2014.

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.

L Becchetti. The behaviour of financial time series: stylised features, theoretical interpretations and proposals for hidden markov model applications. *Speech recognition. Theory and C++ implementation*, 1999.

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.

Mikael Boden. A guide to recurrent neural networks and backpropagation. *the Dallas project*, 2002.

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`.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

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

Jack D Cowan. Discussion: Mcculloch-pitts and related neural nets from 1943 to 1989. *Bulletin of mathematical biology*, 52(1-2):73–97, 1990.

George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2012.

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

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.

John Dines, Junichi Yamagishi, and Simon King. Measuring the gap between hmm-based asr and tts. *IEEE Journal of Selected Topics in Signal Processing*, 4(6): 1046–1058, 2010.

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.

Mark Gales, Steve Young, et al. The application of hidden markov models in speech recognition. *Foundations and Trends® in Signal Processing*, 1(3):195–304, 2008.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR.org, 2017.

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. *Supervised sequence labelling with recurrent neural networks.* Springer, 2014.

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.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 273–278. IEEE, 2013.

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.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696, Sofia, Bulgaria, August 2013. URL `https://kheafield.com/papers/edinburgh/estimate_paper.pdf`.

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.

Herbert Jaeger. *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the" echo state network" approach*, volume 5. GMD-Forschungszentrum Informationstechnik Bonn, 2002.

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`.

Łukasz Kaiser and Samy Bengio. Can active memory replace attention? In *Advances in Neural Information Processing Systems*, pages 3781–3789, 2016.

Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016.

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

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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.

J Lyons. Mel frequency cepstral coefficient (mfcc) tutorial, 2012. URL `http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-freque`

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

Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 1989.

Ian McLoughlin. *Applied speech and audio processing: with Matlab examples*. Cambridge University Press, 2009.

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 Dahl, and Geoffrey Hinton. Deep belief networks for phone recognition. In *Nips workshop on deep learning for speech recognition and related applications*, volume 1, page 39. Vancouver, Canada, 2009.

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.

Vijayaditya Peddinti, TaraN Sainath, Shay Maymon, Bhuvana Ramabhadran, David Nahamoo, and Vaibhava Goel. Deep scattering spectrum with deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 210–214. IEEE, 2014.

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.

Tara N Sainath, Vijayaditya Peddinti, Brian Kingsbury, Petr Fousek, Bhuvana Ramabhadran, and David Nahamoo. Deep scattering spectra with deep neural networks for lvcsr tasks. In *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

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.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Stanley Smith Stevens, John Volkmann, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

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;.

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.

Dong Yu and Li Deng. *AUTOMATIC SPEECH RECOGNITION.* Springer, 2016.

Dong Yu, Li Deng, and George Dahl. Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

Neil Zeghidour, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. A deep scattering spectrum—deep siamese network pipeline for unsupervised acoustic modeling. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 4965–4969. IEEE, 2016.