Master Thesis in Statistics and Machine Learning

Improving Speech Recognition for Arabic language Using Low Amounts of Labeled Data

Mohammed Bakheet

Supervisor: Filip Ekström
Examiner: Jose M. Peña

Division of Statistics and Machine Learning

Department of Computer and Information Science

Linköping University

2021

Abstract

The importance of Automatic Speech Recognition (ASR) Systems, whose job is to generate text from audio, is increasing as the number of applications of these systems is rapidly going up. However, when it comes to training ASR systems, the process is difficult and rather tedious, and that could be attributed to the lack of training data. ASRs require huge amount of annotated training data containing the audio files and the corresponding accurately-written transcript files. This annotated (labeled) training data is very difficult to find for most of the languages, it usually requires people to perform the annotation manually which, apart from the monetary price it costs, is error-prone. A supervised training task is impractical for this scenario.

Arabic language is one of the languages that do not have an abundance of labeled data, which makes its ASR system's accuracy very low compared to other resource-rich languages. In this research, we take advantage of unlabeled voice data by learning general data representations from unlabeled training data (only audio files) in a self-supervised task or pre-training phase. This phase is done by using wav2vec 2.0 framework which masks out input in the latent space and solves a contrastive task. The model is then fine-tuned on a few amount of labeled data. We also exploit models that have been pre-trained on different languages, by using wav2vec 2.0, for the purpose of fine-tuning them on Arabic language by using annotated Arabic data.

We show that using wav2vec 2.0 framework for pre-training on Arabic is considerably time and resource-consuming. It took the model 21.5 days to complete 662 epochs and get a validation accuracy of 58%. Arabic is a right-to-left (rtl) language with many diacritics that indicate how letters should be pronounced, these two features make it difficult for Arabic to fit into these models, as it requires heavy pre-processing for the transcript files. We demonstrate that, we can fine-tune a cross-lingual model, that is trained on raw waveform of speech in multiple languages, on Arabic data and get low word error rate 36.53%. We also prove that by fine-tuning the model parameters we can increase the accuracy, thus, decrease the word error rate from 54.00 % to 47.50 % to 37.06% to 36.53.

Acknowledgements

All praise and thanks due to Allah who blessed me with courage, strength, patience and health, and who let me live to see this thesis through. This publication is part of my research work at Linköping University, funded by a Swedish Institute scholarship.

I would like to express my special appreciation and thanks to my supervisors Filip Ekström, you have been tremendous mentors for me. I also want to thank my examiner Jose M. Pena for his critical feedback and orientation, and our program director Oleg Sysoev for his constant support.

My sincere thanks go to Digitaltolk company for allowing me to conduct this thesis, especially my company supervisor Juan who was a great support during this journey.

Words cannot express how grateful I am to my father for all of the sacrifices that he has made on my behalf
Thank you Father

Thank you my fellow, had it not been for all your prayers and benedictions were it not for your sincere love and help, I would never have completed this thesis. I do thank you all.

Contents

List of Figures

List of Tables

1 Introduction

In the face of a wide diversification of sounds, both experimentally applied and naturally occurring, human speech perception is vigorous. Significant strides are being made regularly to design a machine that mimics the human behavior of speech recognition. Acoustic Speech Recognition (ASR) systems enable machines to recognize human speech and transcribe it into text. There are currently many problems facing ASR Systems, to mention a few: the first one is the speaker problem, and this could be the speaker emotions, gender variation, or age variation. Secondly, the vocal track length could also affect the way people pronounce words. Thirdly, the background and channel noise, this could appear as children noise in a car or in a metro station, or channel noise when speaking over a phone or wireless devices in general. One of the challenging problems is the different accents within the same language (Arabic in this context) could vary depending on the speaker's country of origin.

In the last recent years Automatic Speech Recognition systems have improved the state of the art significantly. Open-source collaborations such as Wav2Vec 2.0 [1] and Mozilla DeepSpeech [2] have brought the speech to text community together to allow almost anyone to train their own models with a comparable performance of the best models 2 years ago with 100 times less data. The problem of data collection is still significant and finding new ways of addressing these issues is critical for the speech to text models of the future.

In the early stages, Hidden Markov Models (HMM) were used as a stochastic approach to solve the problem of speech recognition. HMM achieved good results, but it also had many limitations[3]. A few of these limitations were solved by HMM/Gaussian mixture models (HMM/GMM), in any case, numerous remained unsolved[4]. After that, a hybrid of deep neural networks (DNN), Hidden Markov Models (HMM) was shown to significantly improve ASR performance over the mixture of Gaussian mixture model and Hidden Mrkov Models (GMM-HMM) [5][6].

A problem that appears when using DNN in speech recognition is the large quantities of labeled data that is essential for training neural networks which, in many fields, is more difficult to find than unlabeled data. Thousands of hours of transcribed speech are required to achieve high model accuracy, this large number of hours is unavailable for most of the more than 7000 spoken languages [7]. Learning purely from labeled examples does not resemble language acquisition in humans: infants learn language by listening to adults around them - a process that requires learning good representations of speech [1]. People usually don't transcribe what they say, one must label many hours by themselves or hire someone to do this rather tedious job for them. Even though, this could work in many cases, but we still have the hurdle of spoken vs written languages as some spoken words are difficult to express in a written form. ASR systems have long been relying on labeled data for better performance, so, the more transcribed training data we get the higher the performance. This could mean, for languages with different accents, one must have hours of labeled training data for each accent to acquire high performance in that specific accent. A solution for this problem of labeled data scarcity is to have an unsupervised task where the model is pretrained on huge amount of unlabeled data (which is easier to find), and then finetune the pretrained model on a few hours of labeled data in a supervised task, this solution is refered to as Self-supervised Learning (SSL).

1.1 Background

Per capita, Sweden received the most asylum seekers globally in 2015. Beyond accepting refugees via its longstanding resettlement program, Sweden has taken in rising numbers of asylum seekers since 2012, exceeding 40,000 per year. Most of these recent arrivals have been Syrians, Afghans, and Iraqis [8]. The language barrier has been a big issue for those immigrants when they see a doctor or have a job interview. DigitalTolk is a company that provides interpretation and translation services to public and private sector in Sweden. Patients in emergency rooms, migrants in trauma units, or a job seeker who can't speak Swedish are ought to get the support they need for a better life.

DigitalTolk provides more than 200 languages and dialects support, it has translators and interpreters who speak all those languages. One has to book an interpreter, a person in the current setup, through DigitalTolk's booking system, then the interpreter will be assigned to the client based on an importance score. The current booking system is built on the basis of several importance algorithms, such as the interpreter's interests, experience, competence, or previous feedback from clients.

Even though, the current solution might be efficient for many scenarios, but a faster and more efficient and effective way would be to build an Automatic Speech Recognition (ASR) systems by using the state-of-the-art speech recognition technologies. This new ASR system interprets from Swedish to Arabic and from Arabic to Swedish in real time, it allows non-Arabic-speaking healthcare workers to communicate with patients who are unable to speak Swedish. The ASR does the interpretation through a device that listens to a sentence in a language and then speaks that sentence in another language.

To build an ASR system, a huge amount of training and validation data is needed for the model to achieve high accuracy, and this poses a challenge for building this system as the problem of data annotation (labeled datasets) has been a major issue for ASR systems, and that could be attributed to the fact that individuals generally don't write as much as they speak, and the spoken language tends to be different than the written form of that language. Because of this, a system has to be developed with a few amounts of labeled (annotated) data. Building an ASR with low amounts of annotated Arabic training data will subsequently lay the foundation for building other ASRs for different Arabic dialects or even other languages.

1.2 Research Questions:

This research answers the following questions:

- The problem of data collection is still significant and finding new ways of addressing these issues is critical for the speech to text models of the future. How can we address this problem?
- Can self-supervised learning be used as a technique to learn unlabeled Arabic data that could be used by the ASR? If not, what other techniques could be used?
- Is it possible to achieve high accuracy using Self-Supervised Learning (SSL) of speech representation.

2 Theory

In the face of a wide diversification of sounds, both experimentally applied and naturally occurring, human speech perception is vigorous. Significant strides are being made regularly to design a machine that mimics the human behavior of speech recognition. Over the last few years, advances in both machine learning algorithms and computer hardware have led to more efficient methods for training different speech recognition models.

2.1 How Speech is Digitally Represented

If audio is to be transcribed, going straight to characters is not feasible as characters do not map to different phonemes depending on their pronunciation and position in words. Figure 1 illustrates a wave form and its phonemes representation. A traditional ASR pipeline usually represents words as sequence of "phonemes," For instance: hello = [HH AH L OW]. Phonemes are the perceptually distinct units of sound that distinguish words. Even though it's not clear how fundamental these phonemes are and despite their approximation, but they are somehow standardized and there are different conventions for how to define them. One popular dataset on systems using phonemes is called TIMIT, that has a corpus of audio frames with example of each of the phonemes [9].

Once phoneme representation is acquired, it adds even more complexity to the ASR system, because now the ASR system does not associate audio features with words, it actually associates them with another kind of transcription. which brings us to the introduction of another component into the pipeline that tries to understand how to convert the transcriptions in phonemes into actual spelling, and this is done by using a phonemes dictionary that contains a mapping of each word and its respective phonemes representation.

To transform an acoustic signal to words, it is desired to first explore the elements of speech that are: Segments, Words, Syllables, and Phonemes. Whereas Phonemes are a unique concept in phonetic to recognize words and phones, a speech segment is a discrete unit that can be identified, either physically or auditorily, in the stream of speech. The figure below transcripts the sentence "she just had a baby" with phonemes.

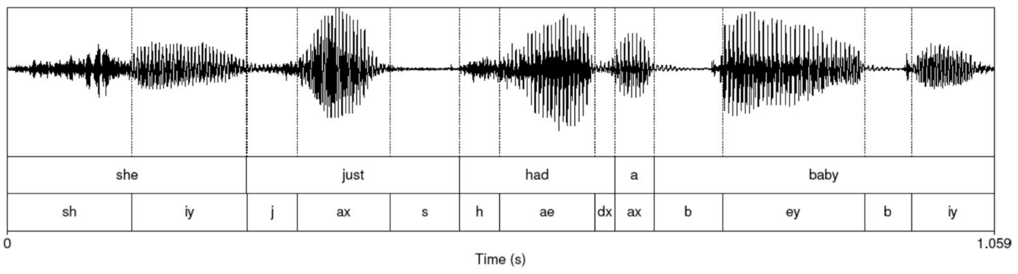


Figure 1: Waveform of someone saying: 'she just had a baby '

Audio is represented in a one-dimensional wave. Typical sample rates for speech: 8KHz, 16KHz, and each sample typically 8-bit or 16-bit. We can think of an audio signal as a one-dimensional vector:

1D vector: $X = [x_1, x_2, ..., x_n]$ representing time and a vector $Y = [y_1, y_2, ..., y_n]$ representing the amplitude of the speech wave y_i at time x_i. However, this representation could change to a totally different form (a spectrogram) that is covered later here Spectrogram in CNNs. This spectrogram could also be used with different models.

2.2 How ASR works

In a traditional ASR system, we would have an acoustic model whose job is to learn the relationship between a sequence of phonemes and the audio we are hearing. It accomplishes that by transforming speech features to probability distribution over phones, $p(O|W)$. For instance, the pronunciation of the letter "t" at the beginning of the word as "tea" and the end of it such as "spirit," so what is going to be the probability of this letter being pronounced? There are different elements involved in ASR systems apart from the acoustic model, that are:

Pronunciation dictionary transforms phones to words. Every word has a number of sounds representing it as in English dictionaries. This could be affected by accent variation. In addition to that, the way someone is speaking is also one of the factors effecting pronunciation (spontaneous speech vs. BBC radio news).
Language model represents a probability distribution over words sequence, $p(w)$. Language models solve the problem of listening without being able to transform sounds to a words or meanings. It provides context to distinguish between words and phrases that sound similar.
Decoder aggregates all knowledge to contract search space, in other word, the most probable word sequence. The job of the decoder is to find the sequence of words (W) that maximizes the probability of a particular sequence (w) given the audio (X).

$$\underset{W}{\operatorname{argmax}} P(W|X) \tag{1}$$

There have been many successful attempts to build acoustic models for each language [10] [11] [12]. However, the research now is to build a language-independent acoustic model, in the current state-of-the-art, speaker independent models are first trained from multiple speakers and then adapted to a specific speaker either before or during recognition. Analogously, language independent modeling is a methodology that combines speech and models from multiple source languages and transforms them for recognition in a specific target language [13]. It's vital to mention that speaking the same language with different accents might affect the ability of that language's ASR system to recognize the speech.

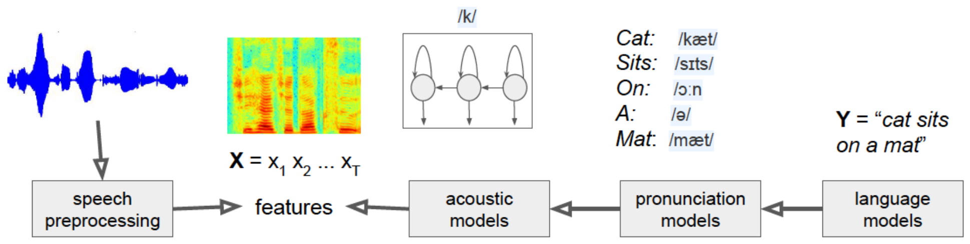


Figure 2: The classic way of building a speech recognition system (https://heartbeat.fritz.ai/)

Figure 2 illustrates the classic way of building speech recognition systems. A generative model of language is built, by producing a certain sequence of words from the language model (rightmost side), and then for each one of the words, the pronunciation model decides how this particular word is spoken by writing in the form of phonemes. The pronunciation models are then fed into an acoustic model which decides how the phoneme sounds. The acoustic model does the description through frames of audio features that is discussed here: "Feature Extraction."

2.3 Speech Features Extraction

Speech signals have high variability for many reasons including gender, accent, emotion, and background noise. To reduce the variability of speech signals some feature extraction should be done. There are many feature extraction techniques used in speech recognition systems including: Linear predictive analysis (LPC), Linear predictive cepstral coefficients (LPCC), Perceptual linear predictive coefficients (PLP), Mel-frequency cepstral coefficients (MFCC), Power spectral analysis (FFT), Mel scale cepstral analysis (MEL), Relative spectra filtering of log domain coefficients (RASTA), and First order derivative (DELTA) [14].

Mel-Frequency Cepstral coefficients (MFCC) is arguably the most commonly used feature for ASR including in DL systems. It is based on short-time analysis (around 20–50 ms), assuming speech to be stationary during that period. MFCC uses mel-scaled frequency weighting and the log function to adopt human auditory systems and compress the feature. The log function is attractive because it is a homomorphic function, and it approximates Gaussian distribution. Then, Discrete Cosine Transformation (DCT) is used to smooth the log spectral energy and de-correlate the components of features. MFCC shows good performances on clean conditions but it is not robust against speech distortions such as background noise, reverberation, channel distortion, etc. One of the reasons is because the log function is very sensitive for low energy spectra where much of speech information resides [15].

2.4 Speech Classification

In ASR systems, structured sequence data is classified, where the label sequences are inferred from the observation sequences. An accurate ASR system is a system that performs with high accuracy in speech recognition tasks by classifying label sequences (sentences) correctly from the observation sequences (wave forms). We can model the posterior probability of the classes (sentences) given the observations in disriminative models, and it is because text data comprises sequences of words with possibly vast number of classes, that the sentence could be broken down into small units (phonemes).

2.4.1 Generative vs. discriminative classifier

The model has to model the distribution of words in the word space. ASR systems infers the label sequences (sentences) from the observation sequences (the acoustic waveform). Speech signals are sequential in nature, and that is where generative models come into view by using Bayes' rule to combine Hidden Markov Model (HMM) acoustic models and N-gram language models. On the contrary, the machine learning and natural language processing (NLP) research areas are increasingly dominated by discriminative approaches, where the class posteriors are directly modeled [16]. It's like when one tells a child to "bring that cup," the child does not actually know unless one points to the cup.

Generative model classification use Bayes' rule e.g., for HMMs, and it can be split into a language model, the prior, $P(w)$ that yields a probability of any sentence, and an acoustic model $p(O_{1:T}, \lambda^{(w)})$ which is the likelihood that a sentence w generated the observations $O_{1:T}$ with model parameters $\lambda^{(w)}$. We can, as a result obtain the posterior by applying Bayes' rule:

$$P(w|O_{1:T}, \lambda) = \frac{p(O_{1:T}|w, \lambda^{(w)})P(w)}{\sum_{\hat{w}} p(O_{1:T}|\hat{w}, \lambda^{(\hat{w})})P(\hat{w})} \tag{2}$$

Whereas discriminative models directly model the posterior given the observation sequence:

$$P(w|O_{1:T}, \alpha) = \frac{1}{Z} \exp(\alpha^T \phi(O_{1:T}, w)) \tag{3}$$

where Z is the normalization constant to ensure a valid probability mass function over all sentences, and α the discriminative model parameters. $\phi(O_{1:T}, w)$ is called the feature function. Feature-function, shown in Figure 3, defines the relationship between the sequence of observations and the sentence label.

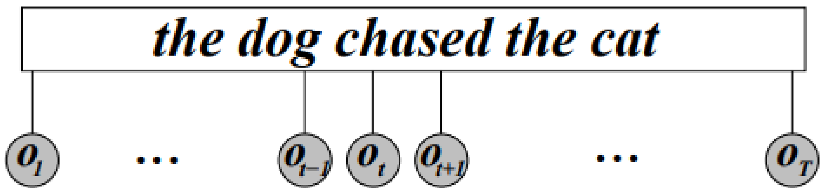


Figure 3: Graphical model for a simple discriminative model [13]

2.5 HMM-GMM in Speech Recognition

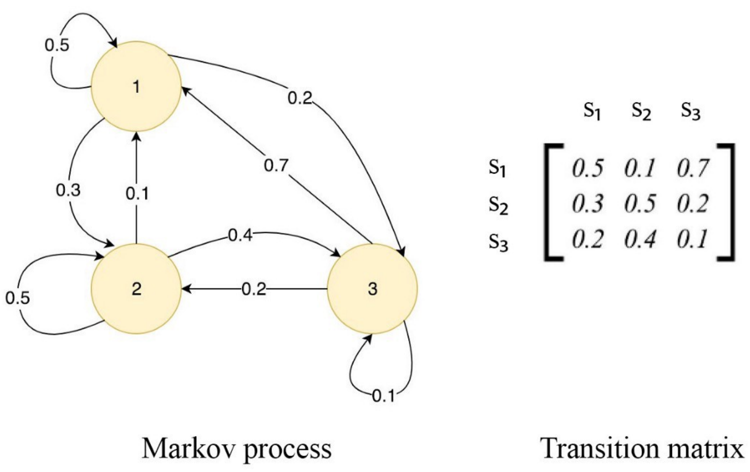
An acoustic model works as a human ear. Nearly four decades ago, the expectation maximization (EM) algorithm was introduced, it is a single advance for training HMMs which is a Markov Chain where the output symbols or probabilistic functions that describe them, are connected either to the states or to the transitions between states [17]. Both Hidden Markov Models (HMM) and Gaussian Mixture Models (GMM) had been, before the era of deep learning, must-learn technologies when dealing with speech signals.

HMMs are used as acoustic models to calculate the likelihood of different classes generating observation sequences (sentences). HMM model consists of hidden variables (States) and observables, a Markov chain also contain the probability of transitioning from one state to another. The modeling part, that is done by HMM, is for transitioning between phonemes and the corresponding observable, and the challenge is to find the optimal sequence of phonemes. Figure 4 shows Markov process and the transitioning between different states.

Gaussian Mixture Models (GMMs) are, on the other hand, used to model the distribution of features for phonemes. They allow to model features with a few possible values, which provides flexibility in speech variations. In Figure 5, it can be seen that when we try to model the phone “sh” using gaussian model, it fixes the utterance of that phoneme and does not allow for speech variations, whereas, when GMM is used for the same purpose, it allows different pronunciations for the same phone, which is more realistic for speech recognition.

2.6 Deep learning in ASR

DNNs have proved more successful in speech recognition than Gaussian Mixture Model (GMM), and that is due to the fact that GMMs are unsuitable for modeling data



$$\begin{array}{c|ccc} & S_1 & S_2 & S_3 \\ \hline S_1 & 0.5 & 0.1 & 0.7 \\ S_2 & 0.3 & 0.5 & 0.2 \\ S_3 & 0.2 & 0.4 & 0.1 \end{array}$$

Figure 4: Markov Process and Transition Matrix

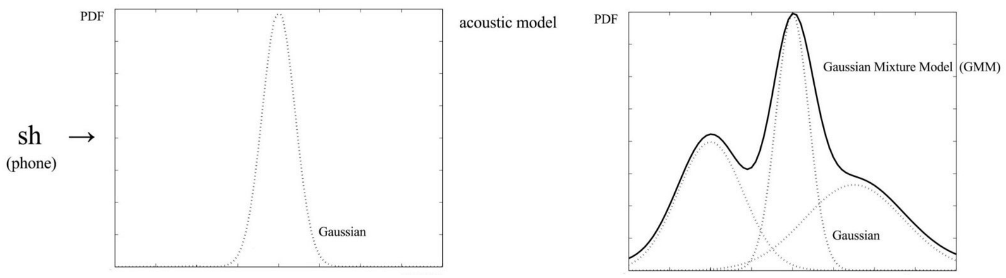


Figure 5: GMM acoustic model

that resides on nonlinear manifolds or near to them. For example, modeling the set of points that lie very close to the surface of a sphere only requires a few parameters using an appropriate model class, but it requires a very large number of diagonal Gaussians or a fairly large number of full-covariance Gaussians [18].

ASR systems rely on modern pipelines that are composed of many processing stages, including specialized input features [19], acoustic models [20], and Hidden Markov Models (HMMs) [21]. Deep learning takes the place of these processing stages and achieves higher performance than traditional methods on hard speech [2]. End-to-end deep learning approach can replace entire pipelines of hand-engineered components with neural networks, end to end learning allows us to handle a diverse variety of speech including noisy environments, accents, and different languages [22]. When deep neural networks are used, a considerable improvement is acquired compared to that of GMM. An acoustic model can get between 10 and 20 percent relative improvement in accuracy [11], which is a huge jump in speech recognition and highly noticeable to a speaker. Achieving this high accuracy by using neural networks is, in some sense, the first generation of deep learning in speech recognition.

The hybrid deep neural network (DNN)- hidden Markov model (HMM) has been shown to significantly improve speech recognition performance over the conventional Gaussian mixture model (GMM)-HMM [23]. This could be attributed to the fact that DNN are capable of modeling complex correlations in speech features.

2.6.1 Convolutional Neural Networks for Speech Recognition

Convolutional Neural Networks (CNNs or ConvNet) are a class of deep neural network, in addition to its common application to analyzing visual imagery, CNNs have many signal processing applications including speech recognition [23]. Rather than using fully connected hidden layers that is similar to DNNs, CNNs present a network structure consisting of *convolutions* and *pooling* layers. The term convolution refers to combining two functions to yield a third function, it is applied in CNNs to the input data by using a kernel or filter to produce a feature map.

$$Q_j = \sigma\Big(\sum_{i=1}^{I} O_i * w_{i,j}\Big) \qquad (j = 1, ..., J), \tag{4}$$

where O_i represents the i-th input feature map, $w_{i,j}$ is each local weight matrix, and σ is the activation function. Both O_i and $w_{i,j}$ are matrices if a two-dimensional feature map is used and vectors when a one-dimensional feature map is used [23].

The limitation of this feature map is seen when a different feature map is produced by a small stride (movement) in the feature position. This limitation is addressed by *down sampling*, and it's applied when important structural elements are included in a lower resolution signal which is generated from the input signal. To resolve this issue, a layer of pooling is added after the convolutional layer. Pooling selects an operation to be applied to the resulting feature maps, the result of this operation is a reduction in size of each feature map. When using CNNs in image processing, the input is a (2-D) array of pixel values at x and y axes, whereas, in speech signals, the input is spectrogram (Figure 6). The temporal behavior of speech signals makes it a fertile area for CNNs application.

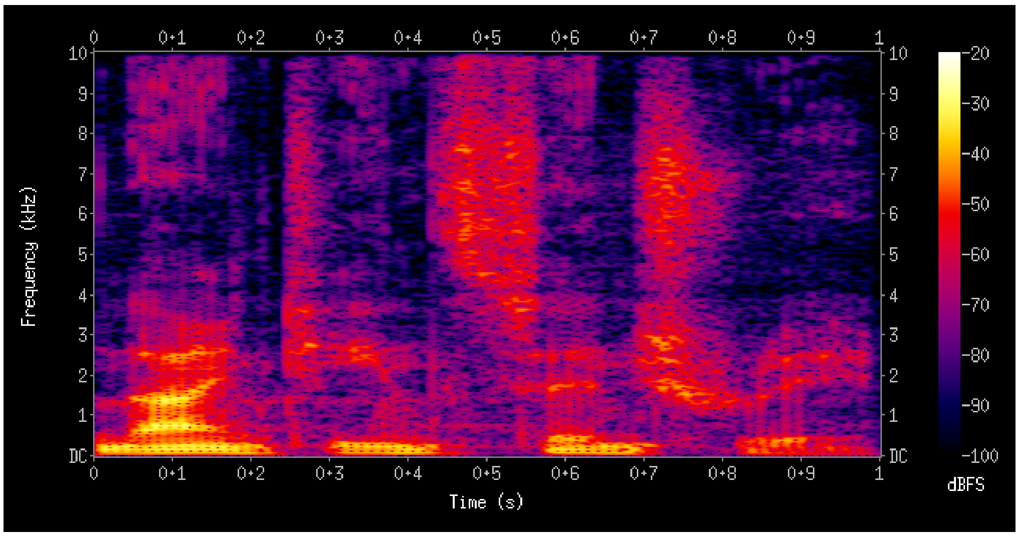


Figure 6: Spectrogram representation of speech that is fed to CNNs

2.6.2 Recurrent Neural Networks (RNNs) for Speech Recognition

An RNN is a class of Artificial Neural Networks (ANNs) that uses sequential data or time series data. RNNs do not assume independence between inputs and outputs, the output of RNNs depends on the memorized prior sequence. For a given input sequence $x = (x_1, ..., x_T)$, a standard recurrent neural network computes the hidden vector sequence $h = (h_1, ..., h_T)$ and outputs vector sequence $y = (y_1, ..., y_T)$ by iterating the following equations from $t = 1$ to T:

$$h_t = \mathcal{H}(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{5}$$

$$y_t = W_{hy}h_t + b_y \tag{6}$$

where the W terms denote weight matrices (e.g. W_{xh} is the input-hidden weight matrix), the b terms denote bias vectors (e.g. b_h is hidden bias vector) and $\mathcal{H}$ is the hidden layer function. $\mathcal{H}$ is usually a sigmoid function, nonetheless, the Long Short-Term Memory (LSTM), which uses purpose-built memory cells to store information, proved to be better at finding and exploiting long range context than sigmoid function [24]. And it is implemented by the following functions:

$$\begin{aligned}
i_t &= \sigma(W_{xi}X_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{7}\\
f_t &= \sigma(W_{xf}X_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{8}\\
f_t &= f_tc_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{9}\\
o_t &= \sigma(W_{xo}X_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{10}\\
h_t &= o_t tanh(c_t) \tag{11}
\end{aligned}$$

where σ is the logistic sigmoid function, and i, f, o and c are input gate, forget gate, output gate and cell activation vectors respectively. Figure 8 illustrates the Bidirectional RNNs (BRNNs) that computes the forward hidden sequence $\overrightarrow{h}$, the backward hidden sequence $\overleftarrow{h}$ and the output sequence y by iterating the backward layer from $t = T$ to 1, the forward layer from $t = 1$ to T and then updating the output layer. If BRNN is combined with LSTM, it results in bidirectional LSTM which accesses both input directions.

$$\begin{aligned}
\overrightarrow{h_t} &= \mathcal{H}(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h_{t-1}} + b_{\overrightarrow{h}}) \tag{12}\\
\overleftarrow{h_t} &= \mathcal{H}(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h_{t+1}} + b_{\overleftarrow{h}}) \tag{13}\\
h_t &= W_{\overrightarrow{h}y}\overrightarrow{h_t} + W_{\overleftarrow{h}y}\overleftarrow{h_t} + b_y \tag{14}
\end{aligned}$$

2.6.3 Transformers in Speech Recognition

The concept of transformers was proposed in 2017 for Natural Language Processing (NLP) [25]. It has, since then, been heavily used to solve different NLP tasks such as text classification and text analysis. Transformers achieved 28.4 Bilingual Evaluation Understudy (BLEU) on the Workshop on Statistical Machine Translation (WMT) dataset 2014, English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU [25]. In machine translation systems, BLUE is a performance evaluation technique that compares the candidate translation, which is done by the machine, and the reference translation or the human-generated translation. For speech recognition, Transformers have achieved competitive recognition accuracy compared to RNN-based counterparts within both end-to-end and hybrid frameworks [26].

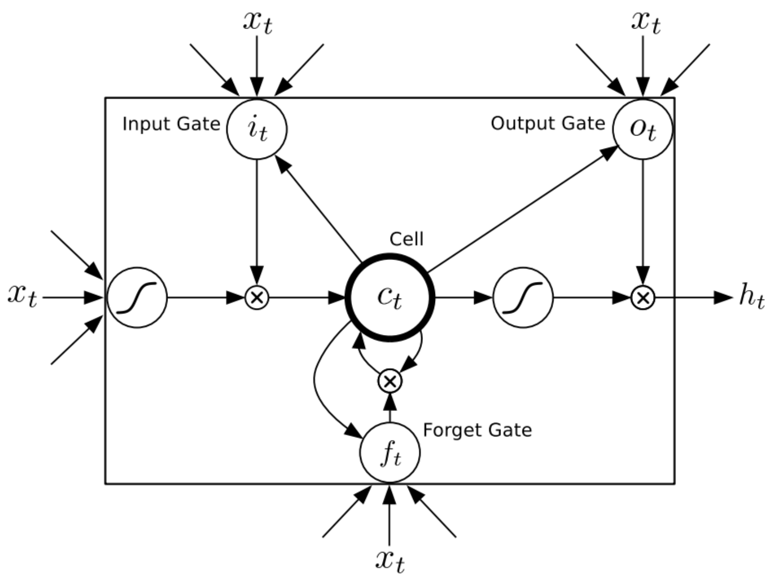


Figure 7: Long Short-term Memory Cell

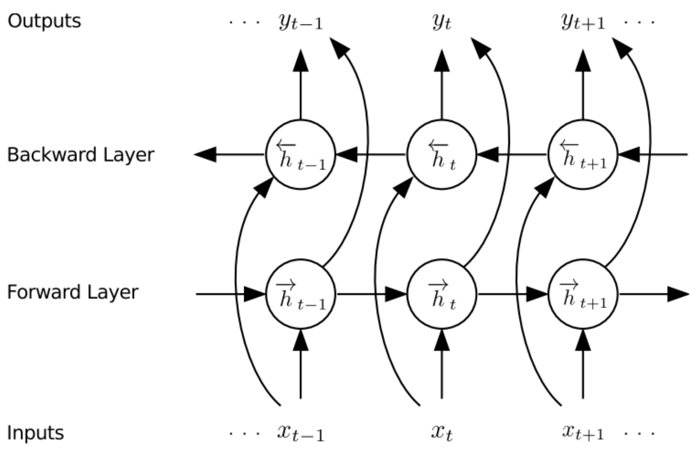


Figure 8: Bidirectional RNN

They introduced an attention mechanism that learns contextual relations between words (or sub-words) in a text. The Transformer, in its most basic form, consists of an encoder and a decoder, the encoder reads the text input by mapping an input sequence of symbol representation $(x_1, ..., x_n)$ to a sequence of continuous representation z = $(z_1, ..., z_n)$ whereas, the decoder produces a prediction for the task by generating an output sequence $(y_1, ..., y_m)$ from the input z it receives from the encoder. As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. This bidirectional characteristic allows the model to learn the context of a word based on all of its surroundings (left and right of the word) [27]. Figure 9 is a high-level description of the Transformer encoder taken from towardsdatascience.com.

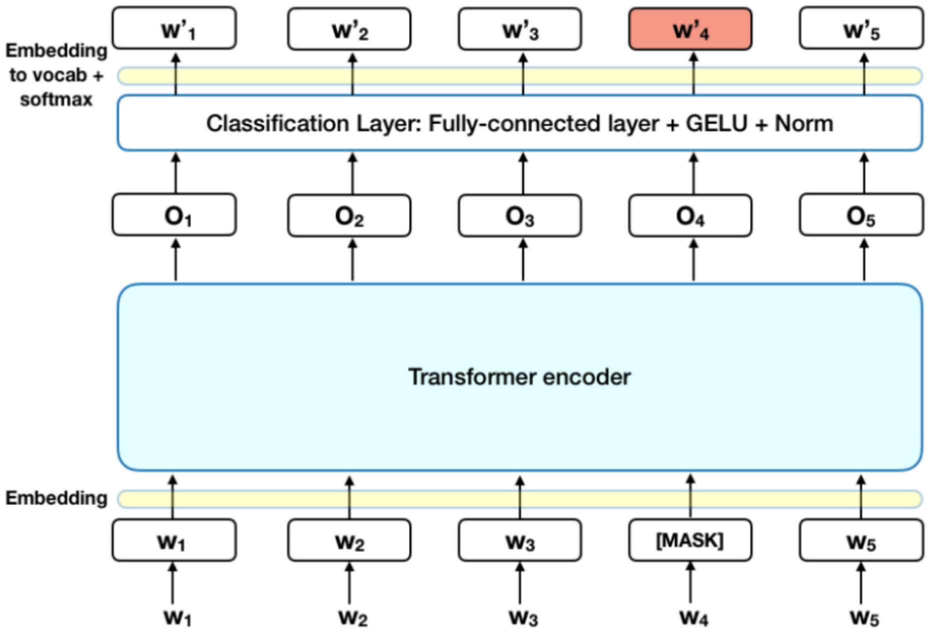

W'1
W'2
W'3
W'4
W'5
Embedding to vocab + softmax
Classification Layer: Fully-connected layer + GELU + Norm
O1
O2
O3
O4
O5
Transformer encoder
Embedding
W1
W2
W3
[MASK]
W5
W1
W2
W3
W4
W5

Figure 9: a high-level description of the Transformer encoder (https://towardsdatascience.com/)

In speech recognition tasks, Transformers can achieve around 6% relative word error rate (WER) reduction compared to the bidirectional LSTM (BLSTM) baseline in the offline fashion, where the entire speech utterance is required as input for the encoder-decoder architecture. On the other hand, in the streaming fashion, where the entire speech is not available, Transformer-XL is comparable to latency-controlled BLSTM (LCBLSTM) with 800 millisecond latency constraint [26]. Transformer-XL was introduced in 2019 for the purpose of enabling learning dependency without the fixed length limitation and by keeping the temporal coherence intact. It does that by by using a segment-level recurrence mechanism and a novel positional encoding scheme [28].

2.7 Self-supervised Learning (SSL)

Neural networks depend on enormous quantities of labeled data to learn, and this scenario doesn't resemble the way humans learn. Self-supervised learning has arisen as a machine learning worldview to solve the problem of labeled data scarcity. It enables learning general data representations from unlabeled examples in a supervised learning task, and fine-tune the model on labeled data [1] by adding an additional predictor to the model that takes in the representations learned by SSL. SSL provides, when pre-training on unlabeled data, effective contextual audio representation. Additionally, by not relying on labels, SSL can avoid many problems related to labels or audio files corruption, thus, increase model robustness. Moreover, self-supervision greatly benefits out-of-distribution detection on difficult, near-distribution outliers, so much so that it exceeds the performance of fully supervised methods [29]. This has been proven successful for natural language processing [30]–[32] and is an active area of research in computer vision [1].

The elemental idea for SSL is to form some auxiliary pre-text task from the input data and then feed it to the model such that while solving the auxiliary task the model learns the underlying structure of the data (object structure in an image). For example, we can use image augmentation to learn a contrastive task by giving similar representations for the augmented versions of the same image and different representation for the augmented versions of different images, thus, the model learns a contrastive task.

2.7.1 SSL in Speech Recognition

A speech file consists of raw audio that could be represented as an audio wave. This wave could, then, be encoded via multi-layer Convolutional Neural Network (CNN), and spans of the resulting latent speech representation are masked. The latent representations are, thereafter, fed to a Transformer network, (discussed in Transformers section), to build contextualized representation by reading the entire sequence of words at once (bidirectionally). The model is then trained via a contrastive task, defined over a quantization of the latent representation, where the true latent is to be distinguished from distractors [1]. This contrastive task requires the true quantized latent speech representation for a masked time step q_i to be distinguished from the other quantized speech representations from other time steps $q_{(1,..,n)-i}$.

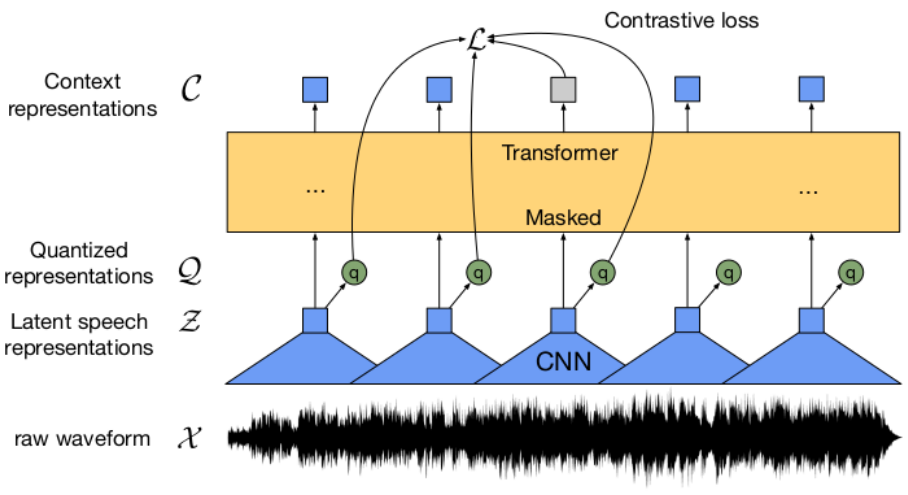


Figure 10: The framework which jointly learns contextualized speech representations and an inventory of discretized speech units. [1]

Figure 10 illustrates how the model learns contextualized speech representation. The model starts with raw audio (a 1-d wave form) that is used to feed a multi-layer CNN which will learn the representations ($\mathcal{Z}$). These representations are vectors representing different wave segments, and are, then, transformed into quantized representations ($\mathcal{Q}$) that are extracted from a codebook. The extraction mechanism depends on the shortest distance between the z vector and q vectors. A Transformer is used to learn contextualized representation by masking out some of the vectors ($\mathcal{Q}$) and measuring the distance between the context ($\mathcal{C}$) and the quantized ($\mathcal{Q}$) representations of those vectors by calculating a contrastive loss.

2.7.2 Contrastive Loss

In its simplest form, contrastive learning means for any positive pairs of input (x_1, x_2), there is a pair of output functions $(f(x_1), f(x_2))$, a neural network function in this context, that are similar to each other. And for a negative input x_3 both $f(x_1)$ and $ f(x_2)$ are dissimilar to $f(x_3)$, Figure 11.

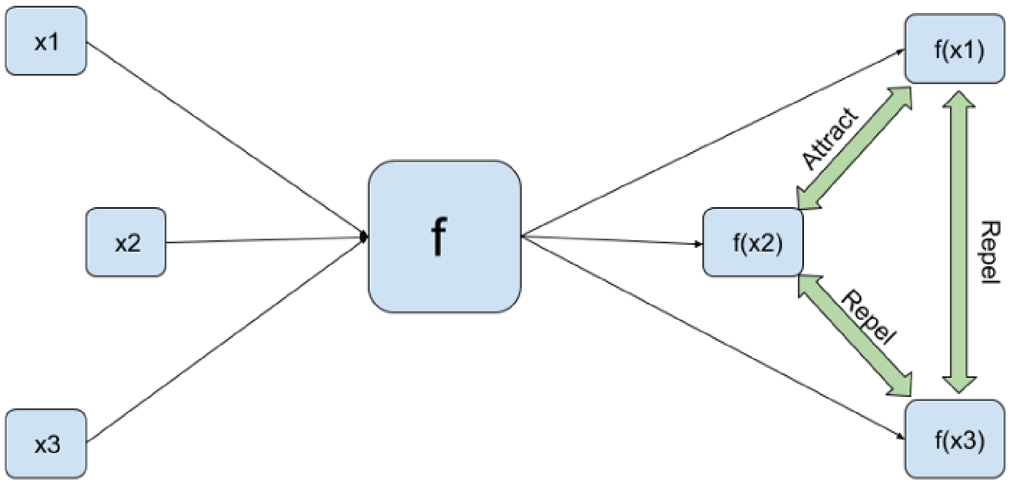


Figure 11: Contrastive Learning (by https://towardsdatascience.com/)

For an output c_t, that is centered over a masked time step t, and, in order for the model to know the true quantized latent speech representation q_t in a set of $K+1$ quantized candidate representations $\tilde{q} \in Q_t$ which includes q_t and K distractors. Distractors are masked time steps of the same utterance, and they are uniformly sampled.

$$\mathcal{L} = -log \frac{\exp(sim(c_t, q_t)/k)}{\sum_{\tilde{q} \sim Q_t} \exp(sim(c_t, \tilde{q})/k)} \tag{15}$$

where

$$sim(c_t, q_t) = \frac{c_t.q_t}{||c_t|| \times ||q_t||} = \frac{\sum_{i=1}^{n} c_{ti}.q_{ti}}{\sqrt{\sum_{i=1}^{n} c_{ti}^2} \times \sqrt{\sum_{i=1}^{n} q_{ti}^2}} \tag{16}$$

$sim(c_t, q_t)$ and $sim(c_t, \tilde{q}_t)$ are the cosine similarity between context representations and quantized latent speech representations [33][34].

2.7.3 Transfer Learning (TL)

Transfer learning is to use a model, that was used for a specific task, for a new related task as a starting point. It stores the knowledge it gained while solving the initial task, and then uses that knowledge for a different but related task. In speech recognition, we can refer to the initial task as “source language” and the new task as “source language,” thus, we can initialize the network for the target language with a pre-trained model from the source language. Different accuracies could be obtained using different TL techniques, 10%-17% relative word error rate reduction with different TL methods over randomly initialized recurrent neural network transducer (RNN-T) model [35].

2.7.4 Pre-trained Models

2.7.4.1 WAV2VEC Pre-training WAV2VEC is a model for unsupervised pre-training tasks in speech recognition developed by Facebook AI. It is a multi-layer convolutional neural network (CNN) that receives raw audio as input and outputs general representations for the raw audio, which can subsequently be used as input to ASR systems. WAV2VEC reduces WER of a strong character-based log-mel filterbank baseline by up to 36 % when only a few hours of transcribed data is available [36]. Figure 12 shows the architecture of WAV2Vec that takes the raw audio as input to a fully connected CNN to solve next time prediction task.

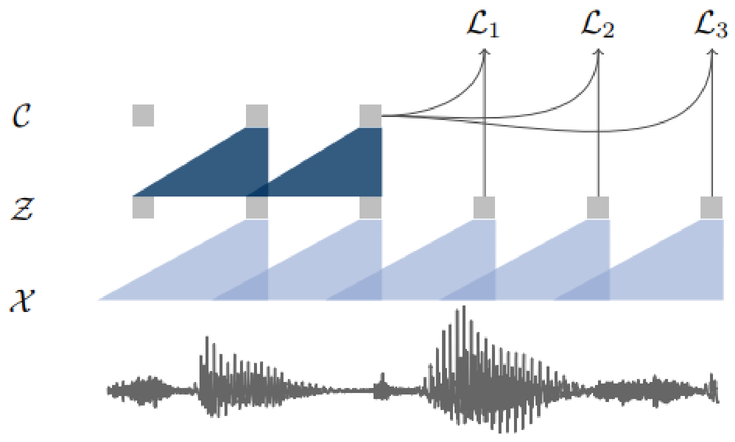


Figure 12: WAV2VEC Fully Convolutional Architecture [35]

The parameterized encoder network $f : \mathcal{X} \mapsto \mathcal{Z}$ is applied on the raw audio samples $x_i \in \mathcal{X}$. With kernel sizes of (10, 8, 4, 4, 4) and strides of (5, 4, 2, 2, 2), The output of the encoder is a low frequency feature representation $\mathrm{z}_i \in \mathcal{Z}$ which encodes about 30 ms of 16 kHz of audio and the striding results in representations z_i every 10ms [36].

The output of the encoder network is then fed into a context network $g : \mathcal{Z} \mapsto \mathcal{C}$ that has nine layers with kernel size three, stride one, and 210 ms receptive field. The context network outputs a single contextualized tensor $c_i = g(\mathrm{z_i}...\mathrm{z_{i-v}})$ from the multiple latent representations $\mathrm{z_i}...\mathrm{z_{i-v}}$ obtained from the encoder network.

2.7.4.2 Cross-lingual Speech Representation (XLSR) As oppose to pretraining speech models on one language (monolingual pretraining), it is also feasible to make use of data from other languages for performance improvement. This has worked well for natural language processing, by achieving state-of-the-art results on cross-lingual classification, unsupervised and supervised machine translation [37]. Learning cross-lingual speech representation in pretraining is not only possible, but it also outperforms monolingual models in many experiments. XLSR showed a relative phoneme error rate reduction of 72% compared to the best known results, and it also improved word error rate by 16% On BABEL corpus compared to a comparable system [38]. This pretrained

XLSR model is finetuned on individual languages which result in a different model for each language. This is particularly useful for languages with low resources as it uses other languages' resources plus the low amount of data from the target language for pre-training, then fine-tune it on the target language.

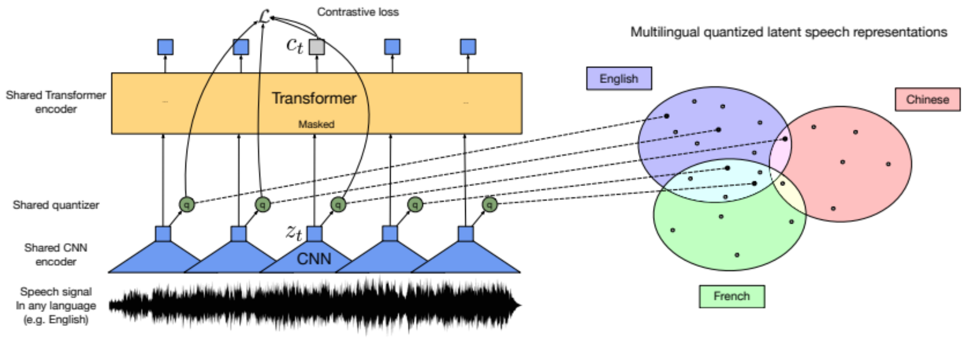

Contrastive loss
$\mathcal{L}$
c_t
Multilingual quantized latent speech representations
Shared Transformer encoder
Transformer
Masked
English
Chinese
Shared quantizer
z_t
Shared CNN encoder
CNN
Speech signal In any language (e.g. English)
French

Figure 13: Cross-lingual Speech Representation (XLSR) Approach [36]

Figure 13 shows the approach of XLSR which is similar to that of WAV2VEC, and that is due to the fact that XLSR is built on top of WAV2VEC framework. Nevertheless, in XLSR architecture, it could be seen that the speech signals of different languages are fed to the shared encoder, which then produces the latent representations $\mathcal{Z}_\sqcup$. The latent representations for all languages are then quantized using a shared quantizer, these quantized representations $q_1 ... q_n$ are then fed to a shared transformer network (encoder) by masking some of the quantized vectors out. The contrastive loss is calculated after that in the same way it is calculated in WAV2VEC, except, in XLSR it is calculated for the output of the shared encoder.

2.8 Connectionist Temporal Classification (CTC)

CTC is an algorithm used to solve the issue of the mismatch between the actual written sentence and the predicted sentence, it uses the history of the target character without assuming conditional independence between characters. In the bottom of Figure 14 we can see how CTC classifies a speech signal, the shaded lines are the output activations, and they correspond to the probability of phonemes matches at particular times. The CTC network predicts only the sequence of phonemes (typically as a series of spikes, separated by 'blanks,' or null predictions), while the framewise network attempts to align them with the manual segmentation (vertical lines) [39].

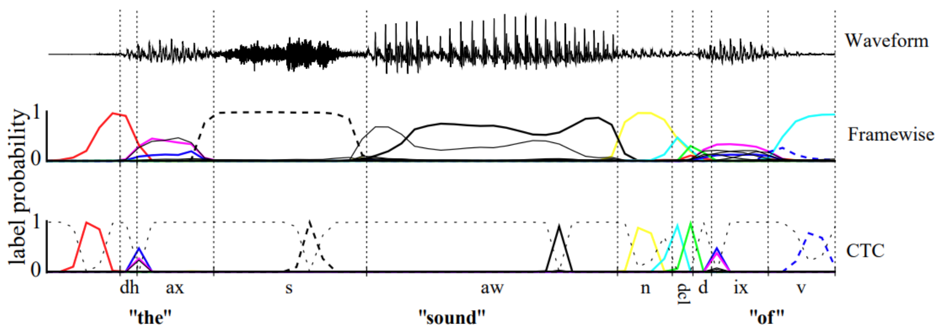


Figure 14: Connectionist Temporal Classification (CTC) [38]

$$p(Y|X) = \sum_{A \in \mathcal{A}_{X,Y}} \prod_{t=1}^{T} p_t(a_t|X) \tag{17}$$

For a given input sequence $X = [x_1, ..., x_T]$ and a corresponding output sequence $Y = [y_1, ..., y_U]$, CTC assigns probabilities for each Y given an X. More specifically, it marginalizes over the set of valid alignments and computes the probability for a single alignment step-by-step.

2.9 ASR Models Evaluation

The error rate of the predicted transcript is calculated as a measurement of how accurate an ASR is. The purpose of this process is to compare the performance of different models and techniques, but it can also go as deep as calculating the performance of these models with different hyper parameters settings. When evaluating an ASR we compare the output of the ASR system with the literal transcription of the input audio. The performance of ASRs depends on a number of factors including audio recording condition (e.g. background noise or the recording channel noise), spoken language variability (e.g. spontaneous speech, formal speech, or different accents or dialects), or the speaker variability (e.g. men and women, a one speaker in different conditions such as illness, different emotions, and tiredness).

There are two key areas related to ASR errors, the first one is the reference-recognised alignment which consist of finding the best word alignment between the reference and the automatic transcription and the second one is the evaluation metrics measuring the performance of the ASR systems [40].

2.9.1 Reference-Recognised Word Sequences Alignment (RRWSA)

RRWSA refers to comparing the referenced and recognised words for the purpose of finding three types of errors. First, substitution error: where a word in the reference word sequence is predicted as a different word. Second, Deletion error: where a word could entirely be deleted in the predicted transcript. The third error type is the Insertion error, and in this error a new word appears in the predicted transcript. This word sequence alignment is normally calculated using Viterbi Edit Distance [41], in which the weighted error score for the word sequence is minimized.

$$d_{\phi}^{v}(x^T, y^V) = -\log \max_{\{z^n : v(z^n) = \langle x^T, y^V \rangle\}} \{p(z^n|\phi)\} \tag{18}$$

The **Viterbi edit distance** calculates the probability of the most likely edit sequence for a string pair (x^T, y^V)

2.9.2 Word Error Rate (WER) Evaluation Metric

Word Error Rate (WER) is the most popular metric for ASR evaluation, it measures the percentage of incorrect words (Substitutions (S), Insertions (I), Deletions (D)) regarding the total number of words processed [40].

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{H + S + D} \tag{19}$$

where S is the total number of words that were substituted in the predicted transcript with other words, D is the total number of words that were deleted in the predicted transcript but they are in the input transcript, I is the total number of newly inserted words into the predicted written words, N is the total number of input words, and H is the total number of correct words.

3 Data

The training data for the Language Model (LM) is limited, and many required Natural Language Processing (NLP) tools such as morph analyzer, diacritizer and text normalizers need to be developed [42]. The data is downloaded from Mozilla Common Voice Corpus (6.1), the size of the dataset is 2 GB with 49 validated hours by the community (the community validates by either matching the voice with the written text or speaking the text itself), 77 hours in total, 672 number of voices, and mp3 audio format.

3.1 How Mozilla Common Voice Community validates

Mozilla depends on contributors who record their voice clips by reading a piece of text and record their voice, this voice is then queued for validation. The clip is valid if another user listens to it and mark it as valid, however, to have the clip in Common Voice dataset, it must be validated by at least two users. If a clip gets a negative vote, it will be returned to the queue, and if it gets another negative vote, then it will be moved to the clip graveyard where all invalid clips exist. [43]

3.2 Data Preprocessing

3.2.1 Preprocessing Audio Files

Since the dataset was initially mp3 audio files, all the files were converted to wav files, the framework takes the following conditions into consideration:

- The sample rate is to be changed to 16000
- The Pulse-code modulation (PMC) is to also be changed to 16 bit
- The silence should be removed from all audio files
- Each audio file should contain only one person speaking

3.2.2 Preprocessing Transcripts

The framework requires the transcript to be in the following format:

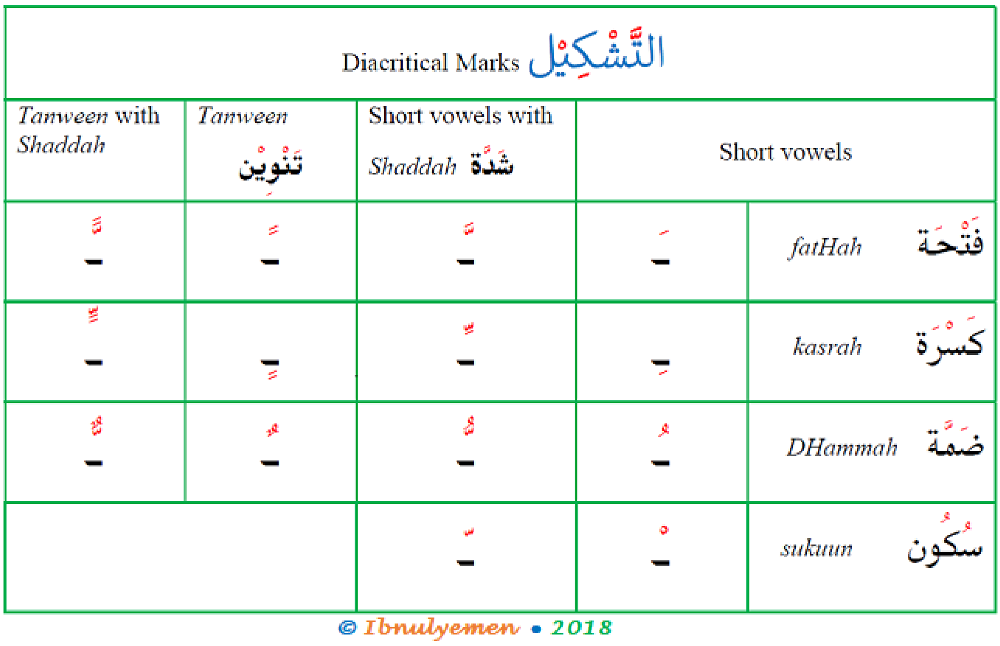
1. One sample per line

- Arabic language is written from right to left (rtl language), hence, the sentences were reversed to meet the criteria for the transcript file.

2. Upper case
   - Arabic language does not have upper and lower case like that of English, therefore, this condition is ignored.
3. All numbers should be transformed into verbal form.
   - All numbers found in the dataset were transformed to verbal form.
4. All special characters (eg. punctuation) should be removed. The final text should contain words only
   - All punctuation marks have been removed from the sentences according to what is required by the framework.
5. Words in a sentence must be separated by whitespace character
   - All words in sentences were separated by whitespaces.

3.2.3 Diacretics

Diacritical marks play a crucial role in meeting the criteria of usability of typographic text, such as: homogeneity, clarity and legibility [44]. Diacritics are placed above, below, or through a letter and they can change the semantic of a word completely, see Figure 15.



Diacritical Marks التَّشْكِيل

Tanween with *Shaddah*	*Tanween* تَنْوِين	Short vowels with *Shaddah* شَدَّة	Short vowels	
ـًّ	ـً	ـَّ	ـَ	*fatHah* فَتْحَة
ـٍّ	ـٍ	ـِّ	ـِ	*kasrah* كَسْرَة
ـٌّ	ـٌ	ـُّ	ـُ	*DHammah* ضَمَّة
		ـّْ	ـْ	*sukuun* سُكُون

© Ibnulyemen • 2018

Figure 15: Diacritical marks in Arabic (https://blogs.transparent.com/arabic/basic-arabic-diacritical-marks/)

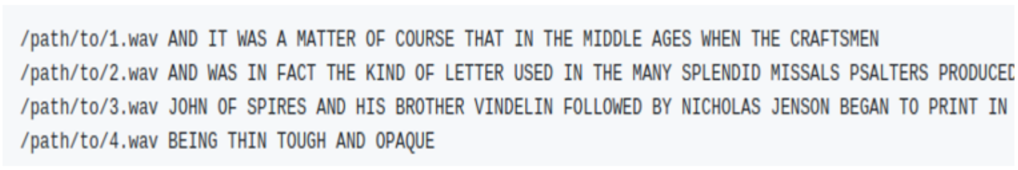
All diacritics were removed from the sentences as they posed a problem when training the model with them, the model read diacritics as individual letters which effected the accuracy since it was required to remove all punctuation marks from the sentences.

4 Methodology

Self-supervised learning is used in this research with its two phases of pre-training and fine-tuning, where transfer learning is the main technique. The implementation is done on a cloud-hosted CPU with Tesla K80 Nvidia GPU, using a wrapper version of wav2vec 2.0 framework [1] that is used for the purpose of training and testing the model. After the data preprocessing phase of Mozilla Common Voice Arabic dataset (6.1), the training and validation datasets are fed to the model for the purpose of fine-tuning, the model is, then, evaluated on the test dataset.

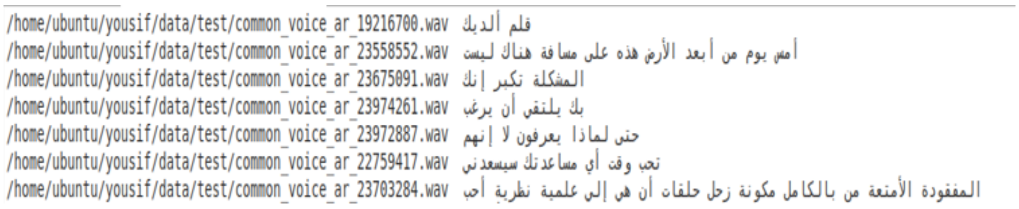
4.1 Data Preparation

The wrapper used for model training was used for English language, and it requires the transcript file to include the path to the wav files and the transcript corresponding to the was file in one line. Since English is a left-to-right (ltr) language and Arabic is a right-to-left (rtl) language, the Arabic sentences are reversed to fit in the wrapper requirements. Figure 16 illustrates the representation of English in the English transcript file, whereas, Figure 17 demonstrates the reversed Arabic sentence in the Arabic transcript file.



```
/path/to/1.wav AND IT WAS A MATTER OF COURSE THAT IN THE MIDDLE AGES WHEN THE CRAFTSMEN
/path/to/2.wav AND WAS IN FACT THE KIND OF LETTER USED IN THE MANY SPLENDID MISSALS PSALTERS PRODUCED
/path/to/3.wav JOHN OF SPIRES AND HIS BROTHER VINDELIN FOLLOWED BY NICHOLAS JENSON BEGAN TO PRINT IN
/path/to/4.wav BEING THIN TOUGH AND OPAQUE
```

Figure 16: a snippet of the English transcript file



```
/home/ubuntu/yousif/data/test/common_voice_ar_19216700.wav قلم ألديك
/home/ubuntu/yousif/data/test/common_voice_ar_23558552.wav أمس يوم من أبعد الأرض هذه على مسافة هناك ليست
/home/ubuntu/yousif/data/test/common_voice_ar_23675091.wav المشكلة تكبر إنك
/home/ubuntu/yousif/data/test/common_voice_ar_23974261.wav بك يلتقي أن يرغب
/home/ubuntu/yousif/data/test/common_voice_ar_23972887.wav حتى لماذا يعرفون لا إنهم
/home/ubuntu/yousif/data/test/common_voice_ar_22759417.wav تحب وقت أي مساعدتك سيسعدني
/home/ubuntu/yousif/data/test/common_voice_ar_23703284.wav المفقودة الأمتعة من بالكامل مكونة زحل حلقات أن هي إلى علمية نظرية أحب
```

Figure 17: a snippet of the Arabic transcript file

4.2 Pre-training Phase

During pre-training, we learn representations of speech audio by solving a contrastive task (Contrastive Loss). The pre-training phase requires having a set of distractors from which to identify the correct quantized latent audio representation. This phase took more than 21 days (504 hours), it was then stopped, after 662 epochs for the purpose of fine-tuning.

$$\frac{p(\mathrm{z_{i+k}}|\mathrm{z_i}...\mathrm{z_{i-r}})}{p(\mathrm{z_{i+k}})} \tag{20}$$

During the pre-training phase, our model predicts future samples from a given signal context, this poses a challenge when modeling the data distribution $p(x)$. In order to avoid this problem, raw speech samples x are encoded into a feature representation z at a lower temporal frequency, and then equation 20 shows the density modeling ratio.

4.3 Fine-tuning Phase

After the pre-training phase, the model is fine-tuned on the labeled test dataset by adding an output layer on top of the Transformer network for prediction. After a warmup for the first 10% of updates, the model is optimized with Adam optimizer. A batch size of 3.2m samples is used.

4.4 Pre-trained Cross-language Model

A second methodology is used in this research, where pretrained models are explored in depth. The pretrained model used is XLSR-Wav2Vec2, the successor to wav2vec, this model learns basic speech units used to tackle a self-supervised task for several languages, and is pretrained and released by Facebook AI Research in September 2020. The pretrained model predicts masked parts of audio, and learns what the speech units should be at the same time. This model can benefit from the availability of some languages and use them for less available languages. XLSR-Wav2Vec2 is fine-tuned using Connectionist Temporal Classification (CTC),

XLSR-Wav2Vec2 model has many parameters to fine-tune, nonetheless, a small number of these parameters, that we fine-tune during fine-tuning phase, is mentioned here:

save__steps is the number of updates steps before two checkpoint saves.
eval__steps is when evaluation is done (and logged) by calculating the WER.
num__train__epochs is the number of full iteration on our data the neural network is going to perform.
per__device__train__batch__size is the batch size per GPU core.
gradient__accumulation__steps is the number of updates steps to accumulate the gradients for, before performing a backward/update pass.
save__total__limit is the number of checkpoint to save.

5 Result

Figure 18 and 19, respectively, show training and validation accuracies during pre-training on training plus validation Mozilla Common Voice datasets. As it could be seen from the two charts, the training accuracy never went above 0.624, whereas, the validation accuracy remained at 0.578. Around epoch 545 the validation accuracy plummeted when the training accuracy kept increasing.

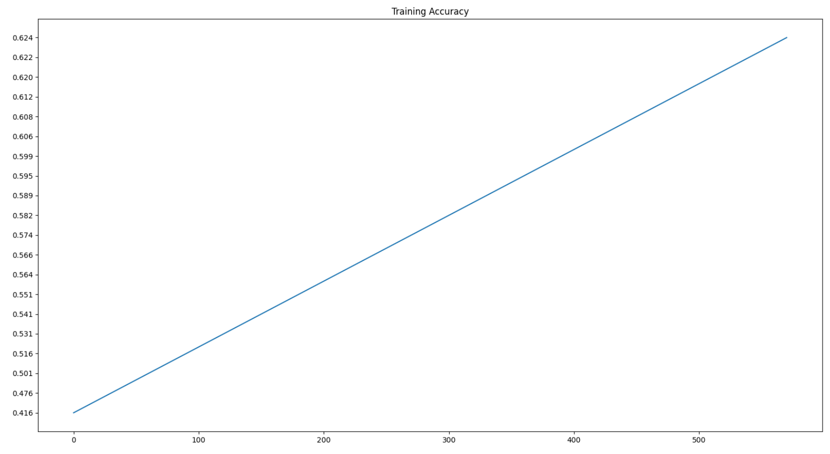


Figure 18: Training Accuracy During Pre-training

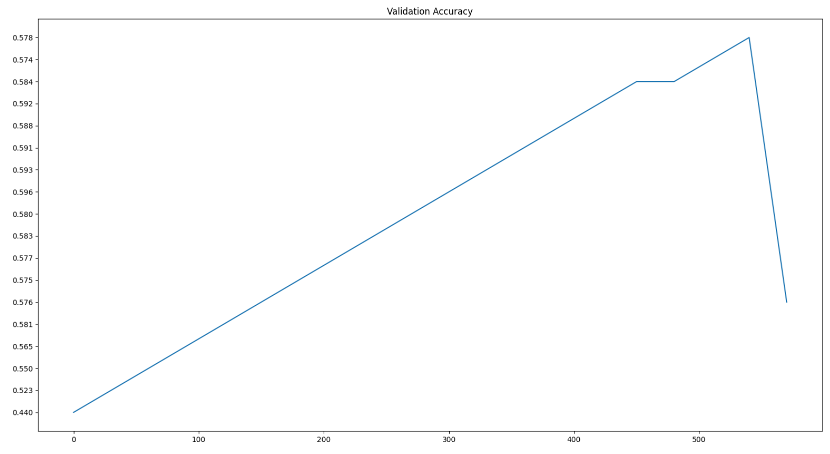
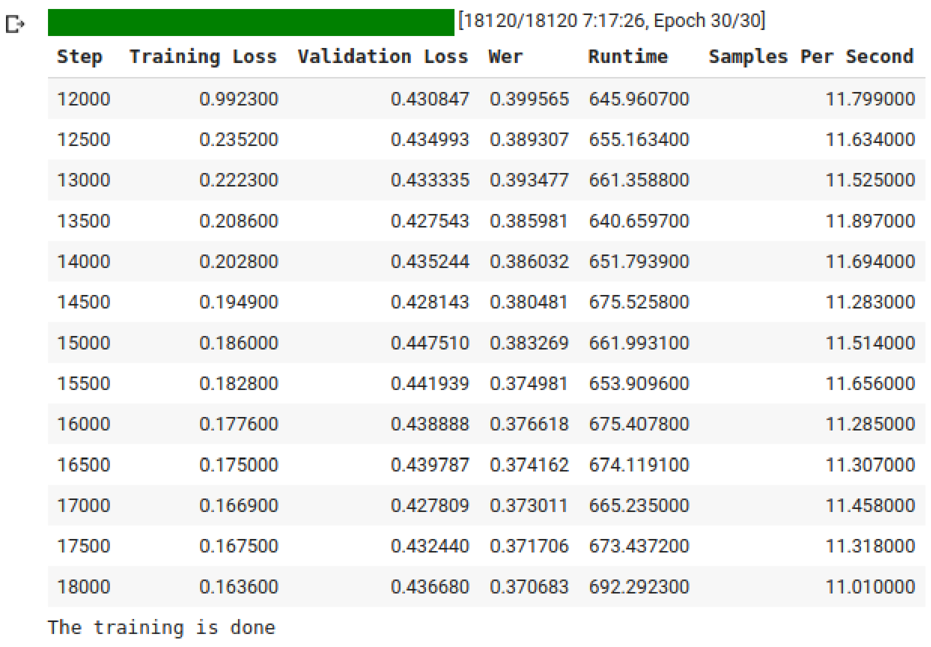


Figure 19: Validation Accuracy During Pre-training

The pre-training phase took more than 21.5 days when it was stopped for the purpose of fine-tuning, there was no early-sopping and the change in validation accuracy was as low as 0.002.

Figure 20 illustrates how WER decrease as the number of epochs and steps increase, it also shows how the training and validation losses decrease, as well as the number of steps.

Figure 21 shows the word error rate with 250 warmup steps, *3e-4* learning rate, batch size of 8, gradient accumulation of 8 steps, and we evaluate the model every 400 steps.

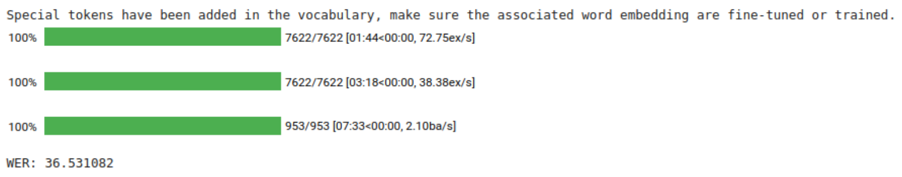


```
[18120/18120 7:17:26, Epoch 30/30]
```

Step	Training Loss	Validation Loss	Wer	Runtime	Samples Per Second
12000	0.992300	0.430847	0.399565	645.960700	11.799000
12500	0.235200	0.434993	0.389307	655.163400	11.634000
13000	0.222300	0.433335	0.393477	661.358800	11.525000
13500	0.208600	0.427543	0.385981	640.659700	11.897000
14000	0.202800	0.435244	0.386032	651.793900	11.694000
14500	0.194900	0.428143	0.380481	675.525800	11.283000
15000	0.186000	0.447510	0.383269	661.993100	11.514000
15500	0.182800	0.441939	0.374981	653.909600	11.656000
16000	0.177600	0.438888	0.376618	675.407800	11.285000
16500	0.175000	0.439787	0.374162	674.119100	11.307000
17000	0.166900	0.427809	0.373011	665.235000	11.458000
17500	0.167500	0.432440	0.371706	673.437200	11.318000
18000	0.163600	0.436680	0.370683	692.292300	11.010000

```
The training is done
```

Figure 20: Training Loss, Validation Loass, and WER



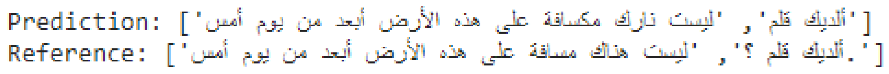
```
Special tokens have been added in the vocabulary, make sure the associated word embedding are fine-tuned or trained.
100%  7622/7622 [01:44<00:00, 72.75ex/s]

100%  7622/7622 [03:18<00:00, 38.38ex/s]

100%  953/953 [07:33<00:00, 2.10ba/s]

WER: 36.531082
```

Figure 21: WER Result for Test Dataset



```
Prediction: ['ألديك قلم', 'ليست ناك مكسافة على هذه الأرض أبعد من يوم أمس']
Reference: ['ألديك قلم ؟', 'ليست هناك مسافة على هذه الأرض أبعد من يوم أمس.']
```

Figure 22: Sample Prediction with 0.365 WER

Figure 22 shows the predicted sentence for one of the audio files from test dataset. As it can clearly be seen in figure 22 the predicted sentence and the source sentence (Reference) are almost the same except for two words. One of the two words is altered by *Insertion* and the second word is changed by both *Deletion* and *Insertion* in terms of WER estimation.

6 Discussion

7 Conclusion

8 Future Work

9 References

[1] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations,” *arXiv e-prints*, p. arXiv:2006.11477, Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.114 77.

[2] A. Hannun *et al.*, “Deep speech: Scaling up end-to-end speech recognition.” 2014, [Online]. Available: http://arxiv.org/abs/1412.5567.

[3] L. R. Rabiner and B. H. Juang, “Hidden markov models for speech recognition — strengths and limitations,” in *Speech recognition and understanding*, 1992, pp. 3–29.

[4] P. Pujol, S. Pol, C. Nadeu, A. Hagen, and H. Bourlard, “Comparison and combination of features in a hybrid HMM/MLP and a HMM/GMM speech recognition system,” *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 1, pp. 14–22, 2005, doi: 10.1109/TSA.2004.834466.

[5] L. Li *et al.*, “Hybrid deep neural network–hidden markov model (DNN-HMM) based speech emotion recognition,” in *2013 humaine association conference on affective computing and intelligent interaction*, 2013, pp. 312–317, doi: 10.1109/ACII.2013.58.

[6] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, “Direct adaptation of hybrid DNN/HMM model for fast speaker adaptation in LVCSR based on speaker code,” in *2014 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2014, pp. 6339–6343, doi: 10.1109/ICASSP.2014.6854824.

[7] G. F. S. Eberhard David M. and C. D. Fennig, “Ethnologue: Languages of the world. Twenty-third edition. Dallas, texas: SIL international,” *Online version: http://www.ethnologue.com*, 2020.

[8] A. Skodo, “Sweden: By turns welcoming and restrictive in its immigration policy,” *Online version: https://www.migrationpolicy.org/article/sweden-turns-welcoming-and-restrictive-its-immigration-policy#:*, 2018.

[9] J. Garofolo *et al.*, “TIMIT acoustic-phonetic continuous speech corpus,” *Linguistic Data Consortium*, Nov. 1992.

[10] O. Enshassi, “Adaptation of acoustic and language model for improving arabic automatic speech recognition,” PhD thesis, 2016.

[11] E. Chang, J.-L. Zhou, S. Di, C. Huang, and K.-F. Lee, “Large vocabulary mandarin speech recognition with different approaches in modeling tones.” Jan. 2000, pp. 983–986.

[12] J. Ivanecký, V. Fischer, and S. Kunzmann, “French–german bilingual acoustic modeling for embedded voice driven applications,” Aug. 2005, pp. 744–744, doi: 10.1007/11551874_30.

[13] W. Byrne *et al.*, “Towards language independent acoustic modeling,” in *2000 IEEE international conference on acoustics, speech, and signal processing. Proceedings (cat. no.00CH37100)*, 2000, vol. 2, pp. II1029–II1032 vol.2, doi: 10.1109/ICASSP.2000.859138.

[14] U. Shrawankar and V. M. Thakare, “Techniques for feature extraction in speech recognition system : A comparative study.” 2013, [Online]. Available: http://arxiv.org/abs/1305.1145.

[15] H. F. Pardede, V. Zilvan, D. Krisnandi, A. Heryana, and R. B. S. Kusumo, “Generalized filter-bank features for robust speech recognition against reverberation,” in *2019 international conference on computer, control, informatics and its applications (IC3INA)*, 2019, pp. 19–24, doi: 10.1109/IC3INA48034.2019.8949593.

[16] M. J. F. Gales, S. Watanabe, and E. Fosler-Lussier, “Structured discriminative models for speech recognition: An overview,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 70–81, 2012, doi: 10.1109/MSP.2012.2207140.

[17] J. M. Baker *et al.*, “Developments and directions in speech recognition and understanding, part 1 [DSP education],” *IEEE Signal Processing Magazine*, vol. 26, no. 3, pp. 75–80, 2009, doi: 10.1109/MSP.2009.932166.

[18] G. Hinton *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012, doi: 10.1109/MSP.2012.2205597.

[19] Jong-Hwan Lee, Ho-Young Jung, Te-Won Lee, and Soo-Young Lee, “Speech feature extraction using independent component analysis,” in *2000 IEEE international conference on acoustics, speech, and signal processing. Proceedings (cat. no.00CH37100)*, 2000, vol. 3, pp. 1631–1634 vol.3, doi: 10.1109/ICASSP.2000.862023.

[20] H. Sak, A. Senior, K. Rao, and F. Beaufays, “Fast and accurate recurrent neural network acoustic models for speech recognition.” 2015, [Online]. Available: http://arxiv.org/abs/1507.06947.

[21] A. V. Nefian, L. Liang, X. Pi, L. Xiaoxiang, C. Mao, and K. Murphy, “A coupled HMM for audio-visual speech recognition,” in *2002 IEEE international conference on acoustics, speech, and signal processing*, 2002, vol. 2, pp. II-2013-II-2016, doi: 10.1109/ICASSP.2002.5745027.

[22] D. Amodei *et al.*, “Deep speech 2: End-to-end speech recognition in english and mandarin.” 2015, [Online]. Available: http://arxiv.org/abs/1512.02595.

[23] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, “Convolutional neural networks for speech recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1533–1545, 2014, doi: 10.1109/TASLP.2014.2339736.

[24] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks.” 2013, [Online]. Available: http://arxiv.org/abs/1303.5778.

[25] A. Vaswani *et al.*, “Attention is all you need.” 2017, [Online]. Available: http://arxiv.org/abs/1706.03762.

[26] L. Lu, C. Liu, J. Li, and Y. Gong, “Exploring transformers for large-scale speech recognition.” 2020, [Online]. Available: http://arxiv.org/abs/2005.09684.

[27] R. Horev, “BERT explained: State of the art language model for NLP.” 2018, [Online]. Available: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.

[28] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, “Transformer-XL: Attentive language models beyond a fixed-length context.” 2019, [Online]. Available: http://arxiv.org/abs/1901.02860.

[29] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, “Using Self-Supervised Learning Can Improve Model Robustness and Uncertainty,” *arXiv e-prints*, p. arXiv:1906.12340, Jun. 2019, [Online]. Available: http://arxiv.org/abs/1906.123 40.

[30] M. E. Peters *et al.*, “Deep contextualized word representations.” 2018, [Online]. Available: http://arxiv.org/abs/1802.05365.

[31] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving language understanding by generative pre-training,” 2018.

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding.” 2019, [Online]. Available: http://arxiv.org/abs/1810.04805.

[33] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning.” 2020, [Online]. Available: http://arxiv.org/abs/1911.05722.

[34] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations.” 2020, [Online]. Available: http://arxiv.org/abs/2002.05709.

[35] V. Joshi, R. Zhao, R. R. Mehta, K. Kumar, and J. Li, “Transfer learning approaches for streaming end-to-end speech recognition system.” 2020, [Online]. Available: http://arxiv.org/abs/2008.05086.

[36] S. Schneider, A. Baevski, R. Collobert, and M. Auli, “wav2vec: Unsupervised pre-training for speech recognition.” 2019, [Online]. Available: http://arxiv.org/abs/1904.05862.

[37] G. Lample and A. Conneau, “Cross-lingual language model pretraining.” 2019, [Online]. Available: http://arxiv.org/abs/1901.07291.

[38] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, “Unsupervised cross-lingual representation learning for speech recognition.” 2020, [Online]. Available: http://arxiv.org/abs/2006.13979.

[39] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks,” in *ICML 2006 - Proceedings of the 23rd International Conference on Machine Learning*, Jan. 2006, vol. 2006, pp. 369–376, doi: 10.1145/1143844.1143891.

[40] R. Errattahi, A. El Hannani, and H. Ouahmane, “Automatic speech recognition errors detection and correction: A review,” *Procedia Computer Science*, vol. 128, pp. 32–37, 2018, doi: https://doi.org/10.1016/j.procs.2018.03.005.

[41] E. S. Ristad and P. N. Yianilos, “Learning string-edit distance,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 5, pp. 522–532, 1998, doi: 10.1109/34.682181.

[42] S. M. Abdou and A. M. Moussa, “Arabic speech recognition: Challenges and state of the art,” in *Computational linguistics, speech and image processing for arabic language*, WORLD SCIENTIFIC, 2018, pp. 1–27.

[43] “Mozilla common voice.” https://commonvoice.mozilla.org/en/about.

[44] M. Hssini and A. Lazrek, “Design of arabic diacritical marks.” 2011, [Online]. Available: http://arxiv.org/abs/1107.4734.