Speech Recognition using 1D Convolution Neural Network

Krishna Sanjaykumar Patel
Student Id: 1111405
Lakehead University

Jay Naimesh Patel
Student Id: 1111407
Lakehead University

Pranav Bipinbhai Bhatt
Student Id: 1109651
Lakehead University

***Abstract*—Speech recognition, as a human-machine interface, plays a very significant part in the area of artificial intelligence (AI). Modern speech recognition systems are a superficial learning system and have drawbacks. This paper focus on implementation of a 1D sequential model using the Convolution Neural Networks (CNNs). It is an alternate form of neural network that can reduce spectral variance and model spectral associations that occur in signals. Besides the literature, Back Propagation is used to train the neural network. Training and testing is performed with a ratio of 80:20. During the entire process, the paper uses an audio data-set as training data and uses the others to check the neural network. Experimental findings demonstrate that CNNs can successfully enforce independent word recognition.**

***Index Terms*—Speech Recognition, Automatic Speech Recognition, Data-set, Natural Language Processing, Machine Learning, Convolution Neural Network, Audio Data-set.**

I. INTRODUCTION

Voice recognition system is a system which is used to convert human voice into signal, which can be understood by the machines. When this is achieved, the machine can be made to work, as desired. The machine could be a computer, a typewriter, or even a robot. Here, only the human is expected to talk. From developing a chat bot to various applications that help us to make our day to day activities easier. The main factor that affects the speech recognition system is the noise which makes it difficult for the system to understand the problem and provide the output. Over recent years, Deep Neural Networks (DNNs) have produced several important advances in the area of speech recognition. Yet we do want a layout that can compensate for minor changes and disruptions in feature space that could be induced by various speech styles and configurations. DNNs can lead to over-fitting and weak generalization in low resource settings. Convolution Neural Networks (CNNs) is an emerging neural network model that aims to resolve some of these issues [1]. The aim of this paper is to implement a model for testing and training over audio dataset using CNNs. CNN is an artificial neural network architecture for multi-stage testing. Compared to DNNs, CNNs have three essential characteristics: local receptive area, weight sharing and sub sampling, which will ensure in-variance of the input goal expression, scaling and distortion to a certain degree [2].

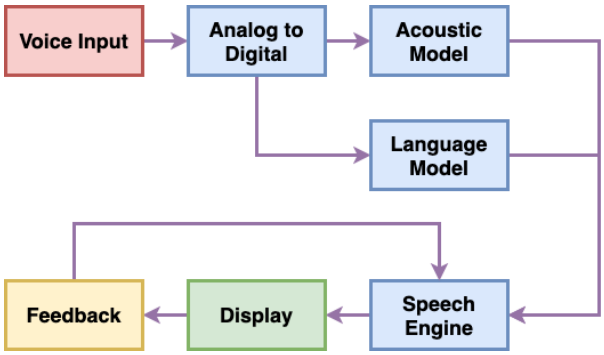

Fig. 1. Block Diagram for Speech Recognition

They often reduce the amount of weights such that the network can be balanced quickly and decreases the chance of over-fitting. In this article, Back Propagation (BP) is used to train the parameters. BP is a supervised learning algorithm that is sometimes used to train multilayer concepts. This is an important way of measuring partial derivatives. It defines the chain law for the derivation of the measure. This consists of forward propagation, back propagation, modification of weight and repetitive repetition. CNN is a multi-layer perceptron built to sense a 2D or 3D signal [3] and [4]. Figure-1 demonstrates the the flow of speech recognition process.

II. LITERATURE REVIEW

Several studies have been performed in the area of speech recognition. For e.g., Morgan [5] performed a speech recognition analysis with the assistance of biased feed-forward networks. The primary aim of the analysis was to shed light on articles that utilised several layers of processing prior to the secret Markov model encoding of word sequences. Throughout the article, several of the methods that combine multiple layers of computation for the purpose of either providing major gains for noisy speech in limited vocabulary tasks or substantial gains for high signal-to-noise ratio (SNR)

speech in broad vocabulary tasks have been identified. In addition, a comprehensive explanation of the methods with architectures that include a large number of layers (depth) and multiple streams utilising Multilayer Perceptron (MLPs) with a large number of secret layers. The analysed paper concluded that, while the deep processing architectures are capable of offering changes in this category, the choice of features and the framework in which they are implemented, including the width of the sheet, may still be significant variables.

L. Deng et al. [6] have been performing a review of Microsoft's research in the field of expression using deep learning since 2009. The paper concentrated on more recent developments that helped shine some light on the various strengths as well as the weaknesses of deep learning in speech recognition. That was achieved by presenting examples of their new research carried out by Microsoft to develop speech-related technologies through the use of deep learning methods. Voice based technologies covered extraction tools, vocabulary processing, acoustic models, voice comprehension, and dialogue prediction. This paper further demonstrates that advances in the design of deep neural networks will be made in order to further develop the acoustic measuring capabilities.

III. Types of Speech Recognition

At a basic level, it can be thought of as speech, that is natural sounding and not rehearsed. An Automatic Speech Recognition (ASR) system with spontaneous speech ability should be able to handle a variety of natural speech features such as words being run together, "ums" and "ahs", and even slight stutters.

A. Isolated Speech

Isolated words usually involve a pause between two utterances, it doesn't mean that, it only accepts a single word, but requires one utterance at a time.

B. Connected Speech

Connected words or connected speech is similar to isolated speech but allows separate utterances with minimal pauses between them.

C. Continuous Speech

Continuous speech allows the user to speak almost naturally and is also called computer dictation

D. Spontaneous Speech

Spontaneous speech is defined in opposition to prepared speech, where utterances contain well-formed sentences close to those found in written documents.

IV. Proposed Model

A. Data-set

The Google Speech Commands Data-set was used for training and testing of speech recognition model developed using 1D Convolution Neural Network in Google Co-Lab. The Google Speech Commands Dataset was produced by TensorFlow and Google Artifical Intellegence teams to demonstrate the illustration of speech recognition utilising the TensorFlow API.

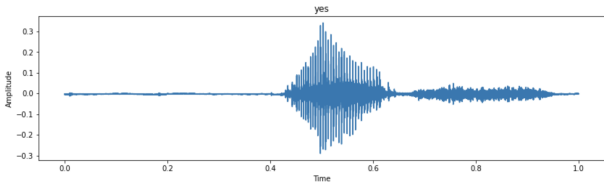

Fig. 2. Sound Frequency for word 'yes'

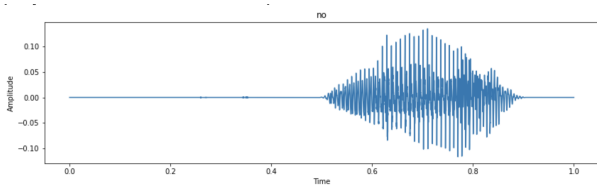

Fig. 3. Sound Frequency for word 'no'

The TensorFlow and Google teams have created the Speech Commands Dataset, and used it to add training and inference sample code to TensorFlow. The dataset has 65,000 one-second long utterances of 30 short words, by thousands of different people, contributed by members of the public through the website. It is released under a Creative Commons BY 4.0 license, and is continued to grow in future releases as more contributions are received. The dataset is basically designed to let you build basic but useful voice interfaces for applications, with common words like "Yes", "No", digits, and directions included. The infrastructure they used to create the data has been open sourced too.

The Google Speech Commands Data-set has audio files of different words such as yes, no, up, down, left, right etc. and the word with their utterances are shown in table-I. and the frequency diagram for the word yes and no are shown in figure-2 and figure-3 respectively.

B. Libraries

In order to perform speech recognition Analysis using 1D-CNN, use of many pre-defined libraries have been done. Keras is used to develop a 1D-CNN model. It is efficient and powerful open source package to develop and evaluate deep learning models. Sklearn is a useful library for machine learning. It

TABLE I
DETAILS OF GOOGLE SPEECH COMMAND DATA-SET

Word	Utterances
Yes	2379
No	2377
Bed	1317
Bird	1731
Cat	1733
Dog	1746
Down	2360
Eight	2353
Five	2358

consists of many efficient tools for statistical modeling and machine learning such as classification, regression, clustering and dimensionality reduction. SciPy is a useful library to solve commonly used tasks in scientific methods such as linear algebra, integration, signal processing. For this proposed model scipy is used for signal processing. It uses Numpy to solve such ambiguity. Librosa is a python package for music and audio manipulation or analysis. It provides certain blocks of information for music retrieval system.

- NumPy
- Keras
- SkLearn
- SciPy
- Librosa
- Tensorflow

C. Data Pre-processing

During the data exploration part earlier, It has been seen that the duration of a few recordings is less than 1 second and the sampling rate is too high. Hence, certain pre-processing steps needs to be done in order to deal with the issue.

Re-sampling is done over the data. Conversion of digital audio file from one sample rate to different sample rate. In the proposed model the sampling rate of the signal is 16,000 Hz. It is re-sampled to 8000 Hz since most of the speech-related frequencies are present at 8000 Hz as per our data-set. We have also removed certain audio files of less than 1 second. The Code Snippet for data pre-processing is available at Appendix VIII(A).

D. 1D-CNN Model

The creation of a CNN is a perfect way to use deep learning for prediction. The Keras Library in Python allows it pretty simple to create CNN. At first, the raw data is converted to NumPy array using NumPy library. Various CNN Layers have been used such as Convolution layer, MaxPooling layer, Flatten layer, Dense layer. Given table-II shows the attributes of respective layers.

Optimizers adjust the weight variables to decrease the error function. Loss feature serves as a reference to the field that informs the optimizer whether it goes in the right direction to hit the bottom of the range, the global minimum. Keras have many optimizer such as AdaDelta, Adagrad, SGD, Adam, Adamax etc. We have tested our model using various of optimizers and best results are taken into consideration along-with the best learning rate. The Code Snippet for proposed model is available at Appendix VIII(B).

TABLE II
CNN LAYERS ATTRIBUTES

Input Layer
Convolution Layer
Conv1D MaxPooling Dropout
Convolution Layer
Conv1D MaxPooling Dropout
Convolution Layer
Conv1D MaxPooling Dropout
Convolution Layer
Conv1D MaxPooling Dropout
Flatten Layer
Dense Layer
Dropout Layer
Dense Layer
Dropout Layer
Dense Layer

E. Saving the model

It is very important to save a model. As it helps to avoid the unnecessary training time. You can resume your model from where you left off. The Code Snippet for saving the model is available at Appendix VIII(C).

V. EXPERIMENTAL ANALYSIS

We compared our model by adjusting different analytical parameters such as batch size, optimizer, learning rate, Epoch, number of layers etc as shown in table-III. Changes in batch size will defer the results to some extent. Which optimizer is used also plays a significant role in making the model more efficient. You may also make changes with the parameter of same optimizer to get a different result.

Learning rate improvements are often considered to render our model more competitive. Moreover, the comparison of testing and training accuracy was done with respect to epoch. The graph generated to generalize the analysis is given in Figure-3.

TABLE III
OPTIMUM ANALYTICAL PARAMETERS

Parameter	Value
Optimizer	Adam
Epoch	10
Batch Size	32
Number of Layer	4
Loss Function	categorical_crossentropy

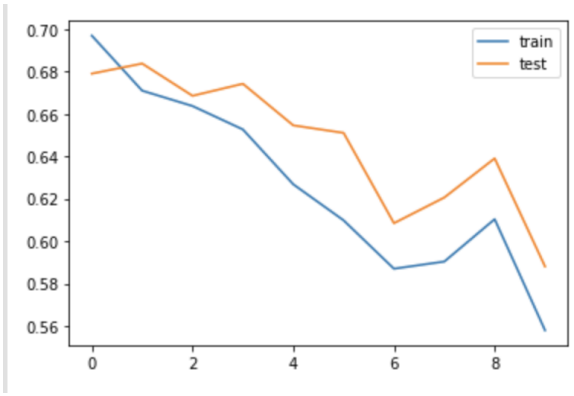

Fig. 4. Train-Test Accuracy Graph

Optimizers are algorithms or methods used to transform batch size outcomes to the maximum accuracy and the lowest L1 loss. Applying the batch size to the optimizers and then measured the performance. After allocating batch size 32, epoch to 10, and the optimizer was Adam, model had the maximum accuracy of 77.8 percent. The relation of Accuracy with respect to batch size can be seen in table-VI and the relation of Accuracy with respect to Optimizer can be seen in table-V.

TABLE IV
COMPARISON OF OPTIMIZER AND BATCH SIZE AND ACCURACY

Batch Size	Loss	Accuracy
32	54	77.8
64	57	73.5
128	68	67

TABLE V
COMPARISON OF OPTIMIZER AND BATCH SIZE AND ACCURACY

Batch Size	Optimizer	Accuracy
32	Adam	77.8
32	AdaDelta	71.5
32	SGD	74.2

The accuracy and L1 loss often depends on the amount of epochs allocated. After carrying out an observational study, It has been concluded that precision decreases as the magnitude of the epochs rises. At the other hand, the value of L1loss should decline as the values of the epochs are raised.

VI. APPLICATIONS

Applications of speech recognition system includes many aspects covering from basic needs to superficial aspects like, used in home automation systems, virtual assistance that can perform tack based on commands, Interactive voice response allows computer to interact with humans vocally, System developed for people having disabilities like hearing aid or partially paralyzed allowing them to type based on their voice commands, automated identification where only authorized entity is allowed to access information about clients, education sector where students with strain injury or disabilities can speak aloud allowing them to be better writers and can increase their fluidity in writing and speaking and vice-a-versa students with low vision or unable to see can have a benefit of hearing the content and act according to it etc.

VII. CONCLUSION

The goal of this paper was to analyze speech recognition using 1D-CNN by training and testing data over specific data-set. Approaches are based on attempting to increase the quality of the test results.

In this paper, a model is developed demonstrating practical implementation of a 1D-CNN has been performed. Model is specifically developed to forecast the word-label after analyzing respective audio data. Various experiments have been carried out by making various changes with certain hyper parameters in the model. As a result of experimental analysis batch size of 32 and 4 convolution layers have been used providing the optimum results. Various optimizers are taken into consideration while developing the CNN model in order to achieve the maximum accuracy and Adam is found to be the best comparatively. As a result, the model provides the results with a Accuracy of 77.82 percent.

TABLE VI
CONTRIBUTION TABLE

Name	Contribution	Task
Krishna	32	Model Accuracy Testing, Report Introduction and Literature Review
Jay	35	Data-set Training and Testing, Pre-processing Data-set, Proposed Model Documentation
Pranav	33	Code Implementation, Error Solving, Report Experimental Analysis and Conclusion

REFERENCES

[1] W Chan, I Lane, "Deep convolutional neural networks for acoustic modeling in low resource languages", Proc. IEEE Int. Conf. Acous. Speech Signal Process. (ICASSP), pp. 2056-2060, 2015.

[2] L. Deng, O. Abdel-Hamid, D. Yu, "A deep convolutional neural network using heterogeneous pooling for trading acoustic invariance with phonetic confusion", Proc. IEEE Int. Conf. Acous. Speech Signal Process. (ICASSP), pp. 6669-6673, 2013.

[3] Zhang Qingqing, Liu Yong, Pan Jielin, Yan Yonghong, "Continuous speech recognition by convolutional neural networks", Chinese Journal of Engineering, vol. 37, no. 9, pp. 1212-1217, 2015.

[4] T. N. Sainath, A. -R. Mohamed, B. Kingsbury et al., "Improvements to deep convolutional neural networks for LVCSR", IEEE Workshop Autom. Speech Recogn. Understand. (ASRU), pp. 315-320, 2013.

[5] N. Morgan, "Deep and wide: Multiple layers in automatic speech recognition," IEEE Trans. Audio, Speech, Lang. Process., vol. 20, no. 1,pp. 7–13, Jan. 2012.

[6] L. Deng et al., "Recent advances in deep learning for speech research at Microsoft," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process.,May 2013, pp. 8604–8608.

VIII. APPENDIX

A. Pre-processing

```
train_audio_path = '/content/drive/My Drive/
    Speechcommand/'

all_wave = []
all_label = []
for label in labels:
 print(label)
 waves = [f for f in os.listdir(train_audio_path+
    label + '/') if f.endswith('.wav')]
 print(os.listdir(train_audio_path+label))
 for wav in waves:
  samples, sample_rate = librosa.load(
    train_audio_path  +'/'+label+'/' +wav,sr =
    16000)
  samples = librosa.resample(samples, sample_rate,
    8000)
    if(len(samples)== 8000):
      print(samples)
      all_wave.append(samples)
      all_label.append(label)
```

Listing 1. Pre-Processing Steps

B. Model

```
inputs = Input(shape=(8000,1))
#First Conv1D layer
conv = Conv1D(8,13, padding='valid', activation='
    relu', strides=1)(inputs)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)
#Second Conv1D layer
conv = Conv1D(16, 11, padding='valid', activation='
    relu', strides=1)(conv)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)
#Third Conv1D layer
conv = Conv1D(32, 9, padding='valid', activation='
    relu', strides=1)(conv)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)
#Fourth Conv1D layer
conv = Conv1D(64, 7, padding='valid', activation='
    relu', strides=1)(conv)
conv = MaxPooling1D(3)(conv)
conv = Dropout(0.3)(conv)
#Flatten layer
conv = Flatten()(conv)
#Dense Layer 1
conv = Dense(256, activation='relu')(conv)
conv = Dropout(0.3)(conv)
#Dense Layer 2
conv = Dense(128, activation='relu')(conv)
conv = Dropout(0.3)(conv)
outputs = Dense(len(labels), activation='softmax')(
    conv)
model = Model(inputs, outputs)
model.summary()
```

Listing 2. Convolution Neural Network

C. Saving the Model

```
model.compile(loss='categorical_crossentropy',
    optimizer='adam',metrics=['accuracy'])
es = EarlyStopping(monitor='val_loss', mode='min',
    verbose=1, patience=10, min_delta=0.0001)
mc = ModelCheckpoint('best_model.hdf5', monitor='
    val_acc', verbose=1, save_best_only=True, mode='
    max')
history=model.fit(x_tr, y_tr ,epochs=10, callbacks=[
    es,mc], batch_size=128, validation_data=(x_val,
    y_val))
```

Listing 3. Saving the Model