Convolutional Neural Networks for Speech Recognition

Essadki Abdelkarim
Lahmadi othmane
othmane.lahmadie@usmba.ac.ma
abdelkarim.essadki@usmba.ac.ma

EL-FAHSI Oussama
Yassine Gannoune
yassine.gannoune@usmba.ac.ma
othmane.lahmadie@usmba.ac.ma

Pr.H.Chougrad
ENSA
USMBA
chougradh@gmail.com

Abstract. The use of a speech recognition model has become extremely important. Speech control has become an important type; our project worked on designing a word-tracking model by applying speech recognition features with deep convolutional neural-learning. Six control words are used (start; stop, forward, backward, right, left). Words from people of different ages. Two equal parts, men and women, contribute to our speech dataset which is used to train and test proposed deep neural networks. Collect data in different places in the street, park, laboratory and market. Words ranged in length from 1 to 1.30 seconds for thirty people. Convolutional Neural Network (CNN) is applied as advanced deep neural networks to classify each word from our pooled data set as a multi-class classification task. The proposed deep neural network returned 97.06% as word classification accuracy with a completely unknown speech sample. CNN is used to train and test our data. Our work has been distinguished from many other papers that often use ready-made and fairly consistent data of the isolated word type. While our data are collected in different noisy environments under different conditions and from two types of speech, isolated word and continuous word.

I. INTRODUCTION

Automatic speech recognition is a technique that involves the use of algorithms and computer programs to convert speech signals into written text. Its primary objective is to enable machines to interpret and act on spoken language. The purpose of automatic speech recognition is to enable computers to receive, interpret, and respond to speech in a readable form. This capability allows for the identification of spoken words and the speaker, as well as emotional recognition. The output of this process can be in the form of written text or other devices that can read specific commands. Our work involved speech signal recognition, which can be categorized into three types based on the signal's characteristics and duration, as illustrated in Figure 1.

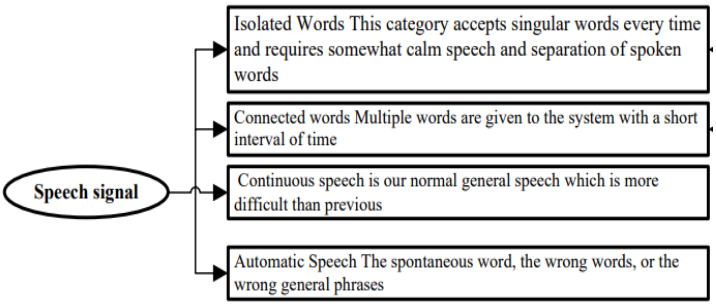


figure 1: The speech types

Those types make this process look like a very challenging task because human speech signals are highly variable due to various speakers' attributes, different speaking styles, and uncertain environmental noises.

Sound waves are a type of wave that travels through a medium via adiabatic compression and decompression. These waves are longitudinal in nature, meaning they vibrate in the same direction as they travel. Spectrograms are used in sound processing to represent signal energy in the form of two-dimensional patterns, with frequency on the vertical axis and time on the horizontal axis. The production of consonants in speech is primarily due to the movement of air in the vocal tract. The process of speech recognition involves several steps, as illustrated in Figure 2.

Figure 2: The general path to completion of speech recognition

The aim of the project is to use the ***C**onvolutional **N**eural **N**etwork* to extract signal features that guide how to convert and classify those features to text.

II. DEEP LEARNING SPEECH RECOGNITION PIPELINE

The automatic speech recognition (ASR) pipeline comprises various components, including a *spectrogram generator* that converts raw audio to spectrograms. An *acoustic model* then takes these spectrograms as input and outputs a matrix of probabilities over characters over time. A *decoder*, optionally coupled with a language model, generates possible sentences from the probability matrix. Finally, a *punctuation and capitalization model* formats the generated text for easier human consumption.

In a typical deep learning pipeline for speech recognition, several components are involved, including ***data preprocessing***, a ***neural acoustic model***, a ***decoder*** optionally coupled with an n-gram language model, and a ***punctuation and capitalization model***. These components work together to analyze the speech signals and generate the desired output in the form of readable text.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

Figure 3 shows an example of a deep learning speech recognition pipeline:

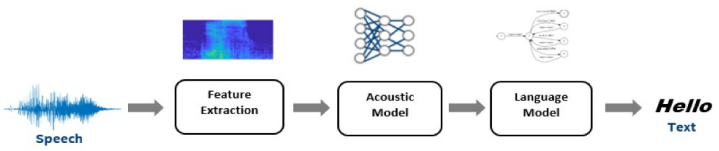


Figure 3: Deep learning speech recognition pipeline

1. ***Models, Methods and Algorithms***

Both *acoustic modeling* and *language modeling* are important parts of modern statistically based speech recognition algorithms.
There are many different algorithms that can be used for automatic speech recognition (ASR). Each algorithm has its own advantages and disadvantages. Some of the most common algorithms include:

- Hidden Markov models (HMMs) are a statistical approach to ASR. They are relatively simple to understand and implement, but they can be computationally expensive.
- Gaussian mixture models (GMMs) are another statistical approach to ASR. They are more accurate than HMMs, but they can be even more computationally expensive.
- Deep neural networks (DNNs) are a more recent approach to ASR. They are capable of achieving state-of-the-art accuracy, but they require a large amount of training data and can be computationally expensive.
- CNNs are a type of DNN that have been shown to be particularly effective for ASR. They are able to learn the temporal structure of speech signals, which is important for accurate recognition.

The main inconvenients of each algorithm are:
HMMs are computationally expensive and can be inaccurate in noisy environments.
GMMs are more accurate than HMMs, but they can be even more computationally expensive.
DNNs require a large amount of training data and can be computationally expensive.

CNNs are a good compromise between accuracy, computational complexity, and the amount of training data required. They are therefore the most commonly used algorithm for ASR today.

In addition to the inconveniences mentioned above, there are a few other challenges that need to be addressed in order to improve the accuracy of ASR systems. These challenges include:

- *Speaker variability:* Different speakers have different voices, which can make it difficult for ASR systems to recognize speech accurately.
- *Environmental noise:* Background noise can also make it difficult for ASR systems to recognize speech accurately.
- *Accents:* People from different parts of the world speak with different accents, which can also make it difficult for ASR systems to recognize speech accurately.

Despite these challenges, ASR systems have made significant progress in recent years. They are now being used in a variety of applications, such as voice assistants, dictation software, and call centers. As ASR technology continues to improve, it is likely to become even more widely used in the future.

2. ***Data Processing***

The availability of high-quality datasets, such as the *LibriSpeech* corpus, has significantly improved the accuracy of speech recognition models. However, it's important to note that data processing plays a crucial role in preparing these datasets for accurate model training.
Data cleaning involves removing any irrelevant or noisy data from the dataset, such as background noise or speech from non-native speakers. Normalization converts the raw audio signal into a format that can be easily processed by the machine learning model, such as Mel-frequency cepstral coefficients (MFCCs). Data augmentation artificially generates new data from the existing dataset, such as adding background noise or varying the speaking rate, to increase the diversity of the training data and improve the generalization ability of the model.

Deep learning models require a large amount of training data to learn and generalize well, as neural networks work similarly to the human brain. The availability of high-quality datasets, such as the LibriSpeech corpus, which contains over 1,000 hours of read English speech, has been instrumental in training accurate speech recognition models using deep learning techniques. These datasets have good compatibility with deep learning techniques, and researchers can leverage them to train state-of-the-art models for automatic speech recognition tasks. Nonetheless, data processing is still an essential step in preparing these datasets for training accurate models.

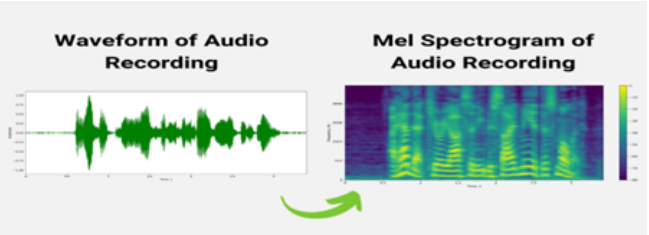


Figure 4: An audio recording raw audio waveform and Mel- spectrogram

We can also use perturbation techniques to augment the training dataset. *Figure 5* represents techniques like noise perturbation and masking being used to increase the size of the training dataset in order to avoid problems like over-fitting.

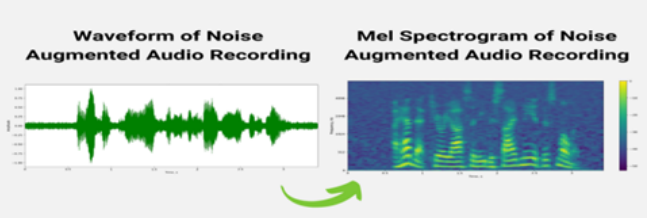


Figure 5: Noise augmented audio waveform to noise augmented Mel-spectrogram image

In speech recognition, one of the main tasks is to extract features from an audio signal that can be used to distinguish between different speakers and recognize the words that they are saying. Spectrograms are one way to extract these features

A spectrogram is a visual representation of the frequency content of a signal over time. By analyzing the frequency components of an audio signal using a spectrogram, we can extract features that are relevant for speech recognition, such as the formants, pitch, and spectral envelope.

the spectrogram tells you how much energy (PSD) there is at a given frequency at a given time

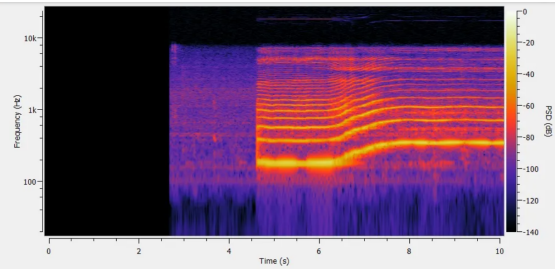


figure 6 : Spectrogram

In the picture above the black region on the left corresponds to the time before the spectrogram was activated. The part with purple and pink haze represents the background noise from my device (a laptop with noisy fans), fridge, wind and other noise that reached the microphone. Then, the orange / yellow lines were produced by pronouncing a vowel. I did a pitch slide, starting with a low pitch (in the middle of the picture) and raising it and then keeping the pitch fixed high.

The spectrogram can be used to distinguish between different speakers by identifying the unique spectral features of their voices. Each speaker has a unique vocal tract, which affects the frequency content of their speech sounds. This can result in differences in the spectral shape of the speech sounds, which can be observed in the spectrogram.

Similarly, the spectrogram can be used to distinguish between different words by identifying the unique spectral features of their phonetic segments. Different words have different phonetic segments, which are characterized by different frequency content and duration. This can result in differences in the spectral shape of the phonetic segments, which can be observed in the spectrogram.

every person in the world has a different spectrogram even if they say the same word. This is because the way that people pronounce words is influenced by a number of factors, including their native language, accent, and vocal tract.

The vocal tract is the passage of air from the lungs to the mouth. It is made up of the throat, mouth, and nose. The shape and size of the vocal tract varies from person to person. This variation in the vocal tract affects the way that people pronounce words.

For example, a person with a long vocal tract will have a different spectrogram for the word "hello" than a person with a short vocal tract. This is because the long vocal tract will amplify different frequencies of sound waves than the short vocal tract.

In addition to the vocal tract, the way that people pronounce words is also influenced by their native language and accent. For example, a person who speaks English with a British accent will have a different spectrogram for the word "hello" than a person who speaks English with an American accent. This is because the British accent and the American accent have different pronunciations for the word "hello."

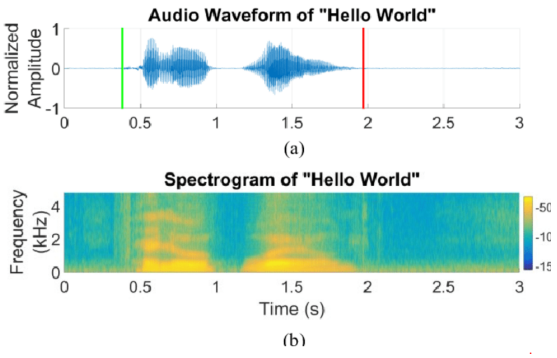
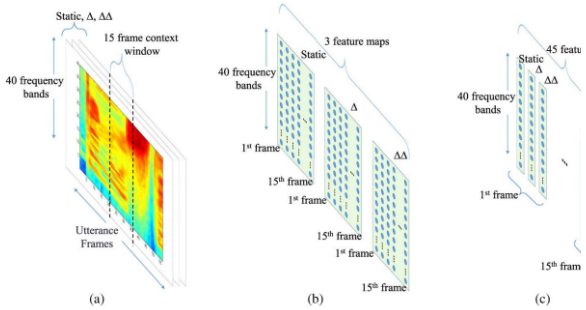


figure 7 :Waveform and Spectrogram of "Hello World"

One common way to extract features from a spectrogram is by using techniques such as MFSC.

The MFSC features capture important information about the spectral envelope of a speech signal or any other audio signal in a compact form.

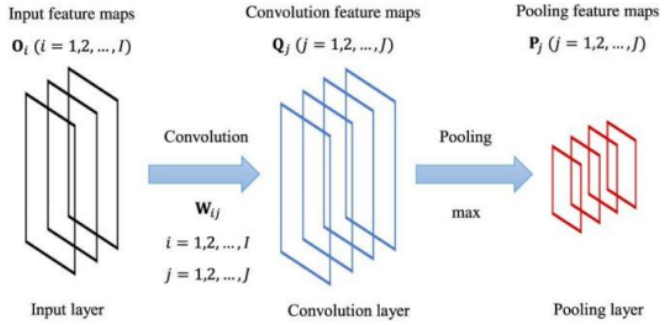


Two different ways can be used to organize speech input features to a CNN. The above example assumes 40 MFSC features plus first and second derivatives with a context window of 15 frames for each speech frame.

There exist several different alternatives to organizing these MFSC features into maps for CNN. First, as shown in Fig. 1(b), they can be arranged as three 2-D feature maps, each of which represents MFSC features (static, delta and delta-delta) distributed along both frequency (using the frequency band index) and time (using the frame number within each context window). In this case, a two-dimensional convolution is performed to normalize both frequency and temporal variations simultaneously. Alternatively, we may only consider normalizing frequency variations. In this case, the same MFSC features are organized as a number of one-dimensional (1-D) feature maps (along the frequency band index), as shown in Fig. 1(c). For example, if the context window contains

15 frames and 40 filter banks are used for each frame, we will construct 45 (i.e., 15 times 3) 1-D feature maps, with each map having 40 dimensions, as shown in Fig. 1(c). As a result, a one-dimensional convolution will be applied along the frequency axis. In this project, we will only focus on this latter arrangement found in Fig. 1(c), a one-dimensional convolution along frequency.

Once the input feature maps are formed, the convolution and pooling layers apply their respective operations to generate the activations of the units in those layers, in sequence, as shown in Fig. 2. Similar to those of the input layer, the units of the convolution and pooling layers can also be organized into maps. In CNN terminology, a pair of convolution and pooling layers in Fig. 2 in succession is usually referred to as one CNN layer



III. CONVOLUTIONAL NEURAL NETWORKS AND THEIR USE IN ASR.

Convolutional Neural Networks (CNNs) are a type of Deep Neural Network (DNN) commonly used in image and video analysis tasks, such as object detection, recognition, and segmentation. CNNs are designed to effectively process spatial data, such as images and videos, by using convolutional layers.The CNN have three main types of layers, which are Convolutional layer ,Pooling layer and Fully-connected (FC) layer .

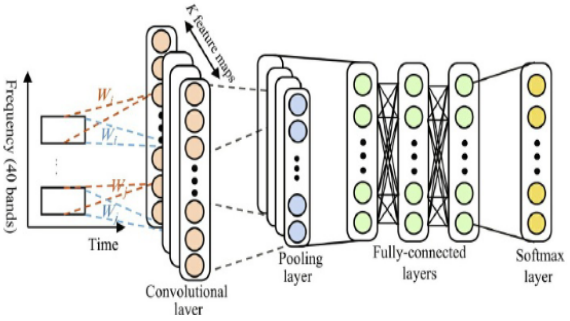


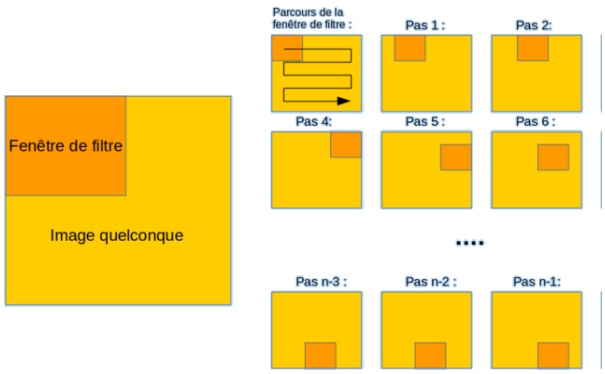
Figure 2: CNN architecture for speech recognition

1. Convolutional Layer :

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs.This layer performs the convolution operation on the input image or feature maps, using a set of learnable filters/kernels to extract spatial features from the input data.Each filter slides across the input data and generates a feature map that highlights the presence of a particular feature. The output of a convolutional layer is a set of feature maps that capture the important spatial information in the input data.The number of neurons in a convolutional layer is

determined by the number of filters used in that layer.

Convolution is a simple mathematical tool that is very widely used for image processing. Convolution is using a 'kernel' to extract certain 'features' from an input image. A kernel is a matrix, which is slid across the image and multiplied with the input such that the output is enhanced in a certain desirable manner. We define a window size that will wander through the entire image. At the very beginning of the convolution, the kernel will be positioned at the very top left of the image then it will shift by a certain number of boxes towards the right and when i arrives at the end of the image, it will shift down one step and so on until the filter has covered the entire image.



In Automatic Speech Recognition ,the input to the CNN is typically a spectrogram or Mel-frequency cepstral coefficients (MFCCs), which are representations of the audio signal in the time-frequency domain and the convolutional layer performs 2D convolutions on the input spectrogram to extract local patterns and features from the time-frequency representation.

2. Pooling layer

The pooling operation is a technique commonly used in convolutional neural networks (CNNs) for reducing the spatial size of the feature maps generated by the convolutional layers.This reduction is achieved by applying a pooling function to several units in a local region of a size determined by a parameter called pooling size.

The pooling operation typically involves dividing the input feature map into non-overlapping or overlapping sub-regions, and then applying a function to each sub-region to obtain a single output value. The most commonly used pooling functions are max pooling and average pooling. In max pooling,as the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling. In average pooling as the filter moves across the input, it calculates the average value within the receptive field to send to the output array. .

The pooling layer helps to reduce the number of parameters and calculations in the network. This improves the efficiency of the network and avoids over-learning.

3. Fully-Connected Layer

In the fully-connected layer, each node in the output layer connects directly to a node in the previous layer. The purpose of the FC layer is to perform a high-level reasoning based on the features extracted by the preceding layers, and to map those features to the final output of the model. In ASR models, the FC layer might be used to map the extracted audio features to the predicted transcript of the spoken words.

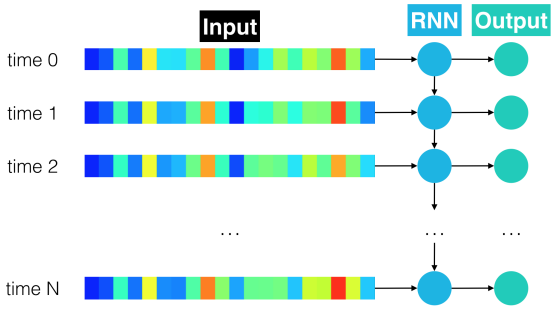
One important consideration in designing the FC layer for ASR is the number of neurons in the layer. The number of neurons in the FC layer is typically large, in order to capture the complex patterns and dependencies in the input features. However, too many neurons can lead to overfitting, while too few neurons can lead to underfitting. Therefore, the number of neurons in the FC layer is often determined empirically through experimentation and validation on a separate dataset.

4- ASR implementation:

the first step in any model implementation is to prepare the data for further training,in our case due to memory and computation limitation,our training was concerned in 2 specific datasets which are:dev-clean and test-clean.the purpose of using these because,all the records are clean from noises which make our model able to use it

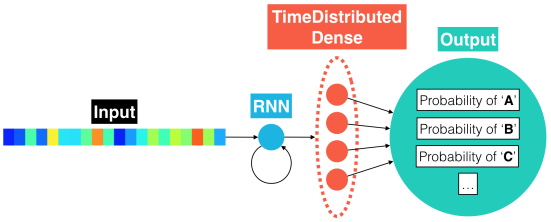
without adding a function to remove unnecessary data in those records,the OpenSlr give all audi files in format of .flac extension(like mp3) which means compressed.In speech recognition, audio data is typically represented as waveform signals in the form of digital audio files such as WAV files(lossless audio format that preserves the original audio quality and),these audio files contain information about the amplitude and frequency of the sound wave over time.in every dataset,they are labels,that are used to train our model we can consider it as output for each input,in our case each directory contain .txt file that contains all transcripts audio files, instead of access each time the same file and read each line which may a headache in training phase,it is a good idea to put all the content in json format.A JSON (JavaScript Object Notation) files are commonly used in ASR to access audio files because they allow us to store metadata about each audio file in a structured format. This metadata may include information such as the transcription of the spoken words, the duration of the audio file, and the sample rate of the audio data. By storing this metadata in a JSON file, we can easily access and manipulate it using programming languages such as Python.
in this article we are going to use 4 models for ASR and we are going to see why CNN alone is not the best best option but instead a combination with CNN and RNN will give good results.Since training models takes more time in our casse from 5-6 hours, it is essential to store the weights and configurations of neural network models used for speech recognition in h5 format which can be loaded and used for further analysis, testing, or deployment of the model. all the results of models used in this project are stored in results folder.After training the model,the final phase is to decoder the decoder is responsible for converting the output of the acoustic model, which is a sequence of phoneme or character probabilities, into the final transcription or sentence.The type of decoder used in our code is the CTC (Connectionist Temporal Classification) decoder. CTC is a widely used decoding method in speech recognition tasks that allows the model to produce variable-length output sequences from variable-length input sequences without requiring alignment between the inputs and the outputs.
the models used to compare why the cnn+rnn are our best option are:

Model 0:Simple RNN:

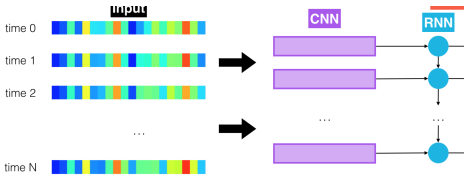


since in english,we have 26 alphabets and with space and comma is 28.The output of the RNN at each time step is a vector of probabilities with 29 entries

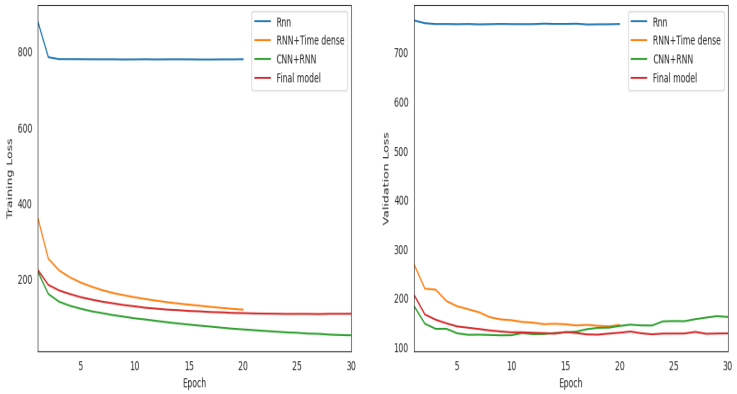
Model 1:Rnn advanced;



Model 2; Cnn+RNN:



Compare the Models;



As we see,in the results,the third model gives good results in terms of training loss and validation loss,that is why we are going to use it for our test to see if he is able to give the content of the records from the test data.

Results:

we tested our model from unseen data and here is the results;

```
/content/drive/MyDrive/LibriSpeech/dev-clean/6295/64301/6295-64301-0002.wav
1/1 [==============================] - 4s 4s/step
[25, 3, 21, 2, 22, 10, 20, 2, 7, 24, 7, 20, 2, 10, 2, 10, 3, 18, 7, 20, 2, 15, 3, 16,
--------------------------------------------------------------------------------
True transcription:

was there ever a happier man than joseph that night as he strode along the footpath
--------------------------------------------------------------------------------
Predicted transcription:

was thr ever h haper man than josoh that ni is e strod alon h fo pat
--------------------------------------------------------------------------------
```

```
1/1 [==============================] - 0s 92ms/step
[22, 10, 7, 16, 27, 10, 7, 20, 2, 20, 3, 21, 10, 6, 17, 16, 21,
------------------------------------------------------------
True transcription:

then he rushed down stairs into the courtyard shouting loudly f
------------------------------------------------------------
Predicted transcription:

thenyher rashdons steers n to lecority ord showding l tly for e
------------------------------------------------------------
```

As demonstrated by our model, it can identify the transcript, but not entirely accurately. However, this is not due to any shortcomings in the model itself, but rather because it was not fed with sufficient data to predict the exact content of each recording.

Conclusion;

In summary, the ASR project aimed to transcribe spoken words in audio files into text using a combination of CNN and RNN models. The CNN model was used to extract features from the audio files, while the RNN model was used to perform transcription. The project faced difficulty in achieving high accuracy due to the complexity of transcribing spoken language, which involves variations in accents, intonation, and speed of speech. The use of a decoder helped to improve the accuracy of the model by enabling the model to decode the output probabilities of the neural network into a sequence of characters. However, the use of only CNN or only RNN models was not sufficient to achieve high accuracy, highlighting the importance of combining different models for ASR. Overall, the project showed the challenges of ASR and the importance of designing complex models to handle the variability and complexity of spoken language.

To conclude:The ASR model achieved good performance, but further improvement can be made by increasing the amount of training data and exploring other architectures.

References;

(1)https://ejle.journals.ekb.eg/article_160440_633339cb02bc485321dd14154e7c3f7b.pdf

(2)https://cramdvoicelessons.blog/spectrograms/#:~:text=The%20spectrogram%20shows%20how%20the,and%20changes%20in%20larynx%20height.

(3)https://www.researchgate.net/profile/Alexander-Hernandez-13/publication/339617157_Convolutional_Neural_Network_for_Automatic_Speech_Recognition_of_Filipino_Language/links/5e5cb752a6fdccbeba1266aa/Convolutional-Neural-Network-for-Automatic-Speech-Recognition-of-Filipino-Language.pdf

(4)https://www.ibm.com/topics/convolutional-neural-networks

(5)https://iopscience.iop.org/article/10.1088/1742-6596/1973/1/012166/pdf