Assignment - II - Digit Recognizer

B117164[1]
WordCount: 2996

I. INTRODUCTION

In recent days, we have seen a steep inclination in Speech processing, synthesis and generation research and the improvement graph had been stagnant for a decade and it has taken a deep ascent after the advent of Graphical Processing units, parallelization and vectorization. Owing to these changes we are now able to generate synthesized user independent speech by Wavenets which are basically diluted ConvNets where convolutions can be trained in parallel by different processing units. This led us to generate speech in a fraction of seconds. But, we still lack in a few areas. We need specific type of data (speech) to generate that kind of a data (speech). Modeling only imitates the data that it has been trained on. Here, in this paper, we will be dealing with a few hypothesis on the available data and those will either be proved or disproved by the conducted experiments.

A. ASR components:

Conventional and State of the art Automatic Speech Recognition models have similar architecture except changes in methods used in different layers. Our objective is fundamentally, given a raw audio the model has to recognize the text (or noise, if trained) spoken. We will cover different levels of recognition and detection in detail in the following sections.

B. Learning models from data:

We live in a world with so much data. All we lack is the proper structure of the data. But this can be parsed and made structured by pre-processing, data transformation techniques. So, machine or deep learning models can either discriminate the data by comparing with each other data or can learn how the data was generated. Hence the name of discriminative and generative models. As we are dealing with speech data, we are facing a non-linear data pattern which was earlier solved and learned by Gaussian Mixture Models. But GMMs have limitations of only modeling fixed state data. Then came Hidden Markov Model which can handle variable length data (vector).

C. Required data:

As far as this assignment is concerned, we need digit utterances. As usual, machine learning models can perform well if the size of training data is increased. It also fits for testing data so that certain models can be regularized or generalized to model exceptional or unseen circumstances. Having said these, we will need two important features in the data. Firstly, large variance in the data i.e., multi-speaker with different age, gender, dialect data. Secondly, accuracy of the labeling should be higher. As the saying goes, 'We can only generate data if that was already trained in the model'. A fair 70:30 ratio between training and testing data is chosen as a rule of thumb.

D. What is hard?

So, the problem is we need data from different kinds and we can only reproduce or recognize data if they have already been fed into the model for training. There are a few circumstances where we have to consider trade-offs between using all the available data or pick the best of the data and by 'best' whats the determining factor. We will be discussing about these in the experiments tried below.

II. THEORY

A. Data Collection and acoustic features

Here, we are dealing with two types of speech recognition model. One, a speaker dependent and an independent model. However the concept and the theory of the models are same but the data collection and feeding. Given raw audio, this layer removes noise and eliminates distortion of channel (less distortion if an exclusive microphone is used and more distortion if a bad microphone that picks up environment noise is used). For the dependent model, I had to record my own voice and label it for both testing and training.

'Like, spectrograms are easy to read consonants and vowels, waveform is not easy to discern text'. So the time based waveform of audio signal is converted into a frequency based form by applying Fast Fourier Transform (FFT). For intuition, we can think of this form as a spectrogram. Hence, the quote above. To get continuous and intricate details, we segment the data into multiple frames typically for 25 ms with 10 ms gaps. This process yield us a format called Periodogram. Then we apply Mel filter banks to get the cepstral coefficients. This process is done to get a rough idea of the distribution of energy across different frequencies. We only keep 12 of the total coefficients as other coefficients do not help much in recognition. As we are in frequency domain, we tried to mimic human perception and implement logarithmic non-linear function that concerns / magnifies (not technically) much about the low frequency data and less about the high frequency data (like a log curve). This is to mimic the mechanism of human ear's cochlea. Now, DCT decorrelates the filtered mel-scaled cepstral values. Data serves better if they are orthogonal and hence this step.

B. Training HMMS

Lets consider the case of connected digits scenario to understand this part. After the audio processing, the HInit module of htk initiates the HMM with the parameters which will later be clustered for a word using K-means algorithm. Then the language model is trained in the HMM model for connected words (digits). Later, the training data's cepstral vectors are fed and the probability score for every transition between the states. Baum-Welch (in HRest) algorithm is used by finding the forward, backward probabilities and by re-estimating the parameters of the data. Then when testing data is fed for testing, Viterbi algorithm helps in determining the maximum P(O/M) - Obesrvation; M- model.

C. Language modeling

We use a plain and simple 1-gram model and a 4-gram model for connected digits. 1-gram model typically is just a dictionary of digits without any other sequential information. For connected speech, we deal with a 4-gram language model where *digit - noise - digit - noise - digit* is the grammar. This developer-defined model will be converted into an FSA by HParse, which in turn will be used by HMM training.

D. Recognition

Given a sentence, the recognizer outputs a series of MFCC sequence which in turn will be used to match with the training data. Given a static or dynamic vector of coefficient features, we have to apply certain machine learning models to find the matching pattern between the dictionary of phonemes and vector mapping and give a confidence score (probability for every possibility of phoneme sequence given the vector feature sequence). Acoustic model has the knowledge of both the given acoustic properties and phonetics properties of a language. So, this is performed using Viterbi algorithm. This uses dynamic programming paradigm to compute all possible paths of the sequence to produce the highest value of sequences and backtrace to produce the sequence.

1) Word prediction: Now, score from Language model and Acoustic model is combined and used to predict the possible word.

III. Experiments

*A. **Meta Data cleaning***

For every machine learning problem, understanding the data is very important. So, first, the information of speakers to decide hypothesis was present as an unstructured file format. And extracting information from it was the first and foremost task which led to the formulate the hypothesis. SO, a python script was written to parse it by universalizing the delimeter between every word. This was later converted to a csv file so that *pandas* can be used to group every type and limitations of the provided data could be visualized by *seaborn* and *matplotlib*. These visualizations which are provided below helped by showing certain groups cannot be used for conducting reliable experiments (or impossible to conduct).

*B. **Speaker Data cleaning (Sanity checking)***

We have only considered the meta data of the speakers but the important data is the speakers audio and labeling data. After deciding the hypothesis from the sorted meta dataset and visualizations from the meta dataset, the speaker datasets are checked for extra/trivial labels like 'ten' label/data. Then for certain speakers training or testing data was not available at all. So these were detected by automating the process of running the whole experiment and the exception throwing speakers are found iteratively. Also, there we some users with few missing resources. So they were spotted out by checking the presence of 30 mfcc files per speaker and one mlf file per speaker (before concatenating all the mlf files). Few wav files for training were also missed. For checking all these exceptions, an exclusive python script was written to figure out. These inquiries were only made for the subset of the speakers who are actually involved in the experiments. Then the speakers were checked for their word error rate as an dependent speech recognizer. These results were also used in the experiments.

*C. **Experiment I - Better data or More data? Whats better?***

For production level speech recognizers, we need a huge amount of data and we cannot complete rely on the data. There are a few checks to make before using it in the model. For a more accurate model, we need more data and as it was already said we cannot trust the data completely even if the data preparation is paid for. So we do need data with good quality too i.e., properly labeled data. So, now we have a dilemma of either using all the data or only pick the high quality data. We can totally relate this scenario to the assignment.

1) Hypothesis:: Better training data with low word error rate will yield higher accuracy and lower WER in the testing of the data.

2) Experiment design:: To prepare the data for this experiment we need to work on a single subset where the data does not vary much in any other case like gender or accent. So it is important that we have to choose a set conforming to these properties and also the data size should be high so that we can fit multiple data size to make better observations. So to choose such a subsample, we need to pick the large samples in each category as follows.

From the figure 1, we can see both male and female are equally distributed. We can choose any one of the genders. The objective is to choose the one with large size so that we can work on different size variations and variety.
From the figure 2, we can see non-native speakers labeled as NN are large in number so we can choose this sample to use in the experiments. UK and SC samples also have a significant size but NN is chosen over the others In figure 3, there were a lot of device types and I could not fit all their names in the y label space of the plot. But from the python output, logitech_speakers were the largely used.

Fig. 1. CountPlot for gender distribution

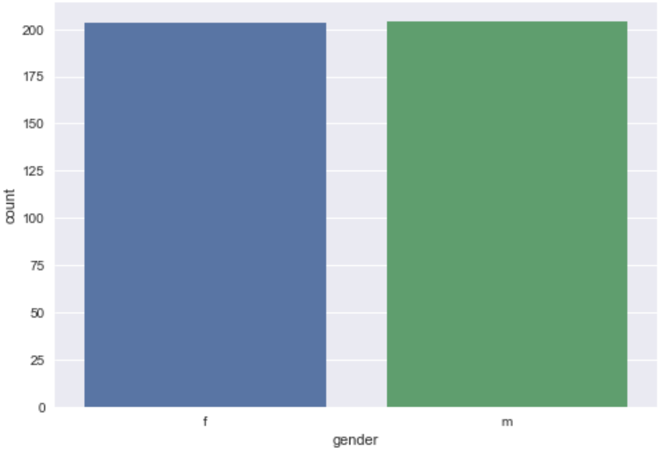
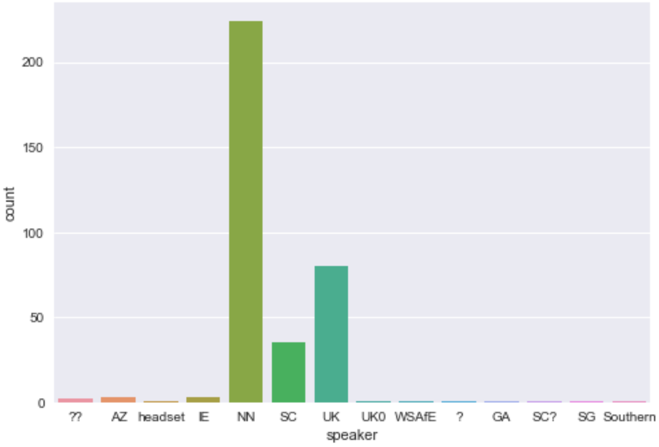


Fig. 2. CountPlot for speakers distribution



From the visualizations, it is obvious that data with gender values as f (female), NN as nativity value and device value as logitech_speakers are larger in number and we can make our experiments on this subset. Also this subset has multiple slabs of WER. We can use this experiment to prove the hypothesis and also we can derive a threshold that we can consider to filter out the training data.

So, the data should be grouped by different accuracy rates so that we can control and vary the value of the accuracy of the users (from dependent speech recognition experiments). Also, the testing data should be constant across different training data and sizes. Preserving this testing data same is very important so that we can rely on different instances of the experiment. Also, the testing data should be containing user data with equal proportion of the accuracy levels (if the training data is accommodating a lower accuracy threshold).

3) Automation and procedures: The experiments proved the hypothesis confirming that data with better quality yield higher overall accuracy rate. Other observations and data insights have also been made from the visualization. The accuracy rate from HResults have been made to written into a text file and a python script was written to parse out the

Fig. 3. CountPlot for device distribution - Ignore Y label**

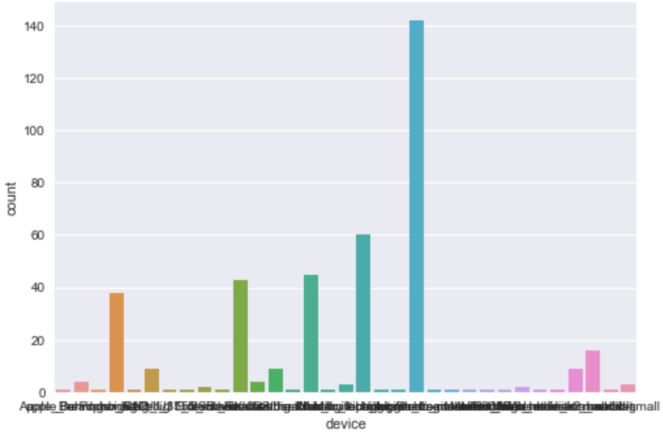
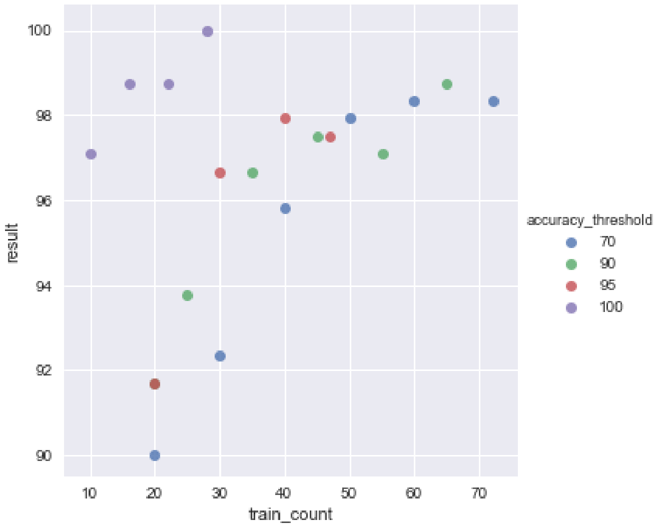


Fig. 4. Plot between training data size and accuracy given the accuracy of the training data as hue



accuracy value and this was written into a csv file in three columns with training data, accuracy rate threshold (hue in the figure) and result accuracy. Later, pandas was used to group the data by threshold values from the csv file and seaborn was used to visualize the countplot as follows.

4) Results and Observations: As we can see from the figure 4, violet data points which represent the threshold of 100% accuracy. This is a strict filter and thats the reason the datapoints have lesser train_count values. Only 28 of the speakers had 100% accuracy. Multiple experiments with different training size were also attempted and as expected, lesser accuracy was observed. Similarly, the training data was then split with different accuracy thresholds. Also these datasets were varied in size to observe the size factor in accuracy. It is true that more data is required but accurately labeled data will give a better accuracy score always. A good quality data in larger size is the scenario that is much desired. Hence, we prove the hypothesis that was made before.

5) Insights and Extra Observations: We can see a linear relation between the observed result accuracy and size of the training data for every colored datapoints. Also, for accuracy 70% (blue colored datapoints) we can see the accuracy level getting saturated after a point (60, form the graph) and continuous as a log curve. This says that irrespective of the size of the dataset, the accuracy of the recognizer gets saturated at a point. So, increasing the quality of the training data is the only way to go and hence this is re-asserting the hypothesis.

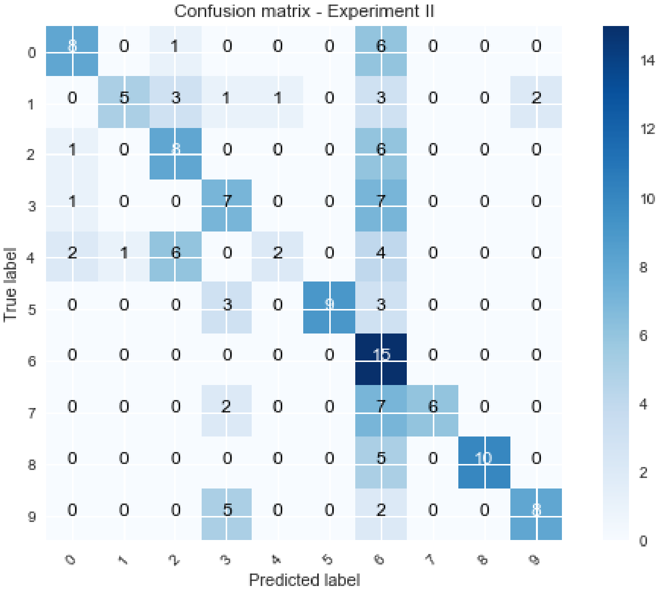
D. *Experiment II - Environment noise affects the accuracy of the model*

1) Hypothesis: Models trained on exclusive microphones (like headphones or headsets) will yield lower accuracy for inbuilt or device inclusive microphones.

2) Experimental design: The speakers with logitech_headsets, apple_mic, headsets were considered as the training set as they represent the exclusive microphone for better voice pick up. And the testing data were the rest of the complete data. The size of the test data was just 40 but the size of the training data was 290. The 70:30 rule was used by hindsight and the size of the training data was truncated to 90. This mixture was chose by equal distribution of accuracy rate.

3) Result: This concern is true with many companies as they have labeled training data with very good microphones but actual real world users may not own such microphones to use the recognizer. As expected the the accuracy for testing data was pretty bad. a 52% was scored on this experiment eventhough the training set contained many 100% accurate speaker data.

Fig. 5. Confusion matrix between labeled and predicted digits



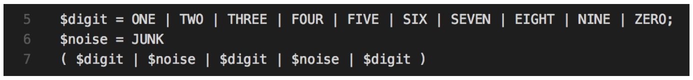
	0	1	2	3	4	5	6	7	8	9
0	8	0	1	0	0	0	6	0	0	0
1	0	5	3	1	1	0	3	0	0	2
2	1	0	8	0	0	0	6	0	0	0
3	1	0	0	7	0	0	7	0	0	0
4	2	1	6	0	2	0	4	0	0	0
5	0	0	0	3	0	9	3	0	0	0
6	0	0	0	0	0	0	15	0	0	0
7	0	0	0	2	0	0	7	6	0	0
8	0	0	0	0	0	0	5	0	10	0
9	0	0	0	5	0	0	2	0	0	8

From figure 5 it is clear that the reason for low accuracy is due to the predicted/recognized label '6'. The reason for ambiguity can be anything. The reasons could be like as they have common pattern of phonemes or /s/ in [six] could be misinterpreted as the noise which usually high frequency fricative noise. This could correlate very much with the word [six]. This is just a theory and as this does not have anything to do with the experiments, they are left here as they have been discussed.

E. *Experiment III - Digit sequences - Grammar complexity*

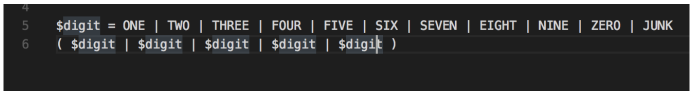
1) Hypothesis: Grammars are important to define and the more sophisticated they are, better the results.

2) Experimental design: For digit sequences, speaker dependent model is followed. For sequential data, the void/noise sounds were labeled as junk in praat and saved accordingly. To make appropriate changes in the grammar file, junk is added in two versions to the grammar file which was later converted to grammar in Markovian format (FST format).



```
$digit = ONE | TWO | THREE | FOUR | FIVE | SIX | SEVEN | EIGHT | NINE | ZERO;
$noise = JUNK
( $digit | $noise | $digit | $noise | $digit )
```

The grammar above is more intricate and has details very specifically indicating the noise labeled separately as JUNK. From the hypothesis this is expected to have higher accuracy.



```
$digit = ONE | TWO | THREE | FOUR | FIVE | SIX | SEVEN | EIGHT | NINE | ZERO | JUNK
( $digit | $digit | $digit | $digit | $digit )
```

This grammar on the other hand is not on point but blunt in specifying the noise as a digit. Here the entropy for $ is higher in comparison with the previous grammar. Hence, the hypothesis. Data was fragmented (in praat) as two noises sandwiched by three digits.

3) Results: As expected, the sophisticated grammar yielded a 96.67 % and the blunt grammar yielded a fair 83.08. The results turned out to prove the hypothesis. %

IV. CONCLUSIONS

In conclusion, most of the hypothesis made are proved and a wide array of hypothesis are considered where one is of the size and quality of the training data, second being a real world speaker variance scenario and third is a sequential extension of data and grammar complexity implications. Other exploratory analysis were also made. For example, pruning variation was done by varying the factor value between 60 and 120 and a deeper knowledge of viterbi strategy of HMM algorithm was conceived using HVite. They are not added here as asked.

REFERENCES

[1] YU, DONG. Basic Architecture of ASR Systems. AUTOMATIC SPEECH RECOGNITION, SPRINGER LONDON LTD, 2016.

[2] Stuttle, Matthew Nicholas. Hidden Markov Models for Speech Recognition. A Gaussian Mixture Model Spectral Representation for Speech Recognition, University of Cambridge, 2003, pp. 644.

[3] University, Cambridge. HTK Toolkit. HTK Speech Recognition Toolkit, htk.eng.cam.ac.uk/docs/docs.shtml.