Accents, Noise, and ASR Models

Divinity Gines
Professor Will Styler
12 June 2024
LIGN 168 Spring 2024

Supplementary Materials: Additional audio files referenced are in this dataset The github is at https://github.com/divinityg/lign168_finalproj
Feedback: I would like specific feedback on if it is plausible to look at the different language families and how it could have affected the WER

Self-Assessment

Cover Page and Self-Assessment: Masterful. I've completed a cover page that has everything needed and completed the assessment.

Scope of Writeup: Masterful. I believe that I have explained the different results of my findings and how it relates to my hypothesis.

Demonstration of Knowledge: Masterful-to-Acceptable. I've discussed different key concepts from class in regards to ASR systems, but I feel that I didn't touch on too many different aspects outside of this scope.

Richness: Masterful. I go through the code and comment the different steps that I finished.

Formatting and Length: Masterful. This is a 2,500 word paper that covers what needs to be said. I also used a reasonable formatting convention

Structure and Organization: Masterful. All of the sections are clearly labeled with transitions as needed.

Language: Masterful. I have reviewed the paper and edited it accordingly.

Academic Integrity: Masterful. I have referenced and cited all of the resources I have used for this paper.

Proposed Project Grade: 92% I think this is a fair grade given how much effort I have put into this project. This project was time consuming due to the amount of data I worked with initially, but I was able to condense it in order to follow the guideline and structure it accordingly. Because some elements are missing and I wasn't able to complete the full original dataset, I believe that a 92% is a fair grade.

Abstract

Everyone has an accent. Whether they acknowledge it or not, depending on their native language and where they grew up, people develop different accents. During my time at UC San Diego, I have come across a wide variety of different people with different accents. Some were easier to understand than others. The human brain had trouble interpreting some types of accented English. With my knowledge of how computers recognize language, their simplistic knowledge of language has led me to question how well they would be able to interpret people's speech of different accents. In this paper we will explore two different automatic speech recognition (ASR) models: whisper and wav2vec; and see how well they are able to interpret accented English even when mixed with noise.

Introduction

To assess the computing processing of language, I decided to compare two different ASR models: whisper and wav2vec. In addition to testing two different ASR models, I wanted to see how well they were able to interpret the original audio in comparison to audio mixed with pure white noise. The white noise was added in to test the models language identification. As mentioned earlier, everyone has an accent, but for the sake of this experiment "accented" English will refer to non-American or British English. Because accented English can already be hard to understand, adding another layer would help test the models' language identification. It would also be interesting to see any minute differences between the models as Whisper was trained on almost 709 times more hours worth of data than wav2vec. Whisper was trained on 680,000 hours and wav2vec was trained on 960 hours [2][4]. This was inspired by discussion 3 in LIGN 168 where we tried to break different ASR models and calculated it through the word error rate (WER). The experiment will involve the two models processing the data and outputting it into a csv file where it can be further analyzed based on the WER. If we test the different models with the original audio and noise added audio, then Whisper should be more accurate due to the amount of hours put into training the model.

My Approach

Dataset

I used an open source data set from the Speech Accent Archive that has 2140 speech samples with 214 different native languages [7]. In each sound file the speaker will read the following passage in English:

"Please call Stella. Ask her to bring these things with her from the store: Six spoons of fresh snow peas, five thick slabs of blue cheese, and maybe a snack for her brother Bob. We also need a small plastic snake and a big toy frog for the kids. She can scoop these things into three red bags, and we will go meet her Wednesday at the train station."

Model

In order to compare against a wide variety of accents while also acknowledging the fact that this dataset is open source, I decided to use two different automatic speech recognition systems: Whisper and Wav2Vec. The two models are some of the top ASR models right now. They were developed by OpenAI and Facebook respectively. On top of this, I will also be comparing how well each model handles noise. Each audio will be mixed with pure noise in order to make it harder for the model to discern words.

Word Error Rate

The word error rate (WER) of each audio processed by the models will be calculated with the python package jiwer. The word error rate is calculated as follows:

$$WER = \frac{S_w + D_w + I_w}{N_w}$$

S the number of substitutions, I is the number of insertions, and D is the number of deletions that have occurred in the reference in comparison to the predicted sentence [3]. Whereas N is the total number of words in the reference.

Experimental Setup

Device Specifications

The device that I used for this experiment was a Microsoft Surface Pro 8 with the following computer specifications: Intel Core i5 and 8GB RAM. The code can run offline, meaning everything was ran locally through the device.

Data

The audio used was from an open source dataset where all participants have already given their consent to be involved in the project. Because all of the audio was saved as an mp3, I first had to convert them to wav format. I did this using Audacity.

Initially wanting to work on the entire 2140 speech samples, I realized that my computing power limited my ability to process data in a timely manner. This meant that processing the audio files through the two different models would take a substantial amount of time. For Whisper, my device ran for 5 hours in order to process all 2140 speech samples. For Wav2Vec, my device was projected to run 6 hours but unfortunately my laptop restarted before it could finish processing all speech samples. I will go over this more in the results section.

This meant that I had to pivot in order to finish the experiment in a timely manner. I decided to use the first speech sample of each native language in order to process audio files faster.

After organizing the files so that only the first speech sample of each native language was saved to a folder, I began to mix each speech audio file with the pure noise file.

Environment

The following packages had to be installed in order to run the code: whisper, transformers, torch, librosa, os, csv, and jiwer. The libraries whisper, transformers, transformers, torch, and librosa were used for the ASR models. The libraries os and csv were used in order to write into a csv file to make sure the data was organized into clean tables. Finally, jiwer was used to calculate the WER of the audio files and predicted transcriptions.
The following code was used to run the Whisper model [1]:

```
#Transcribe using whisper
def transcribe_whisper(audio_path):
    model = whisper.load_model("base")
    result = model.transcribe(audio_path)
    return result['text']
```

For wav2vec I used the following code [6]:

```
processor =
Wav2Vec2Processor.from_pretrained("facebook/wav2vec2-large-960h-lv60-self"
)
model =
Wav2Vec2ForCTC.from_pretrained("facebook/wav2vec2-large-960h-lv60-self")

#Transcribe using wav2vec
def transcribe_wav2vec(audio_path):
    speech, rate = librosa.load(audio_path, sr=16000)
    input_values = processor(speech, return_tensors="pt",
sampling_rate=16000).input_values
    with torch.no_grad():
        logits = model(input_values).logits
    predicted_ids = torch.argmax(logits, dim=-1)
    transcription = processor.batch_decode(predicted_ids)[0]
    return transcription
```

Here we have the code for the creation of the csv file as well as conducting WER using jiwer [3]:

```
#Double \\ is for Windows OS
original = "data\\original"
noise = "data\\noise"
```

```
csv_file_path = "masterfile.csv"
reference = "Please call Stella.  Ask her to bring these things with her
from the store:  Six spoons of fresh snow peas, five thick slabs of blue
cheese, and maybe a snack for her brother Bob.  We also need a small
plastic snake and a big toy frog for the kids.  She can scoop these things
into three red bags, and we will go meet her Wednesday at the train
station."

#Normalizes transcription
transforms = jiwer.Compose(
    [
        jiwer.ExpandCommonEnglishContractions(),
        jiwer.RemoveEmptyStrings(),
        jiwer.ToLowerCase(),
        jiwer.RemoveMultipleSpaces(),
        jiwer.Strip(),
        jiwer.RemovePunctuation(),
        jiwer.ReduceToListOfListOfWords(),
    ]
)

#Writes CSV file
with open(csv_file_path, mode='w', newline='', encoding='utf-8') as file:
    writer = csv.writer(file)
    #Headers
    writer.writerow(['Filename',
'Whisper','WER_Whisper','Wav2Vec','WER_Wav2Vec','Noise_Whisper','WER_Noise
_Whisper','Noise_Wav2Vec','WER_Noise_Wav2Vec'])

    #Loop through each file in the directory
    for filename in os.listdir(original):
        if filename.endswith(".wav"):
            #Audio file paths
            original_path = os.path.join(original, filename)
            noise_path = os.path.join(noise, filename)

            #Clean audio data whisper transcription
            whisper_transcription = transcribe_whisper(original_path)
            whisper_wer = jiwer.wer(
                            reference,
```

```
                            whisper_transcription,
                            truth_transform=transforms,
                            hypothesis_transform=transforms,
                        )

            #Clean audio data wav2vec transcription
            wav2vec_transcription = transcribe_wav2vec(original_path)
            wav2vec_wer = jiwer.wer(
                            reference,
                            wav2vec_transcription,
                            truth_transform=transforms,
                            hypothesis_transform=transforms,
                        )

            #Noisy audio data whisper transcription
            whisper_noise_transcription = transcribe_whisper(noise_path)
            whisper_noise_wer = jiwer.wer(
                            reference,
                            whisper_noise_transcription,
                            truth_transform=transforms,
                            hypothesis_transform=transforms,
               )

            #Noisy audio data wav2vec transcription
            wav2vec_noise_transcription = transcribe_wav2vec(noise_path)
            wav2vec_noise_wer = jiwer.wer(
                            reference,
                            wav2vec_noise_transcription,
                            truth_transform=transforms,
                            hypothesis_transform=transforms,
                        )

            #Write in to CSV
            writer.writerow([filename,
whisper_transcription,whisper_wer,wav2vec_transcription,wav2vec_wer,whispe
r_noise_transcription,whisper_noise_wer,wav2vec_noise_transcription,wav2ve
c_noise_wer])
            print(f"Processed {filename}")
```

Dataset Organization

Once all of the data was collected into one csv file, I decided to open the CSV file in excel in order to work with the data. I found it much easier to work with the dataset in the Excel file rather than in pandas as a dataframe. It allowed me to work faster without having to worry about how to code it to work properly. In it I highlighted where the WER was 0% or 100%. I also highlighted where the WER difference between the clean and noisy audio was negative, signifying that the model did better with the noisy data in comparison to the clean data.

Results

Loading the models with the entire dataset was very time consuming and computing heavy on my device. Whisper finished the entire dataset in 5 hours whereas wav2vec was still running at 5 hours and crashed before it could finish. This could have been due to how wav2vec required a larger amount of computing power.

Some key differences I observed between the two models is the language output. Whisper had 15 transcriptions where it included non-Roman alphabet characters. Wav2vec on the other hand had all of the transcriptions in English. This is the most likely reason why in the clean audio transcriptions Whisper had a higher average WER than wav2vec as seen in Table 1. For wav2vec, the language output was all capital letters with no punctuation. This allows for a smoother transition between different technology applications for WER, but it is less human friendly because of its lack of punctuation and imitation of human language.

Average WER	Whisper	wav2vec
Clean	23.3043%	12.5507%
With noise	31.8188%	77.7174%
Difference between noise and original audio	8.5145%	65.1667%

Table 1: The average WER based on three different variables

With clean data, Whisper had a 100% WER for 8/200 of the different native languages. Wav2vec had 0/200 with 100% WER. Additionally, Whisper had 7/200 transcriptions with 0% WER and wav2vec had 18/200. This allowed for wav2vec to have a better transcription ratio than Whisper.

An interesting thing that I observed was that out of the transcriptions that had 0% WER, 5 of those transcriptions had a 0% WER for both models. Their native languages were: Chamorro (Austronesian Family), Croatian (Indo-European Family), English (Indo-European Family), Frisian (Indo-European Family), and Shona (Niger-Congo Family). English was part of the 0% WER which follows my own intuition, and 2 other languages stemmed from the same family of Indo-European. For Chamorro, even though it is an Austronesian family, the speaker is 18 years old at the time and that would mean that Saipan, their birthplace, was under U.S. rule. So the U.S. could have influenced the territory to speak English more. Shona, a Niger-Congo language, was a surprising native language to have 0% WER when it is not one of the Indo-European languages.

Although the native English speaker audio resulted in 0% WER in clean audio data across both models, it wasn't without error in noise mixed data, nor did it have the lowest WER for both models. This is surprising to see but could be due to the volume level of the speaker in comparison to the white noise.

Surprisingly, Whisper did not do drastically terrible with the noise added audio in comparison to the clean audio. In fact, this time there were only 11 Whisper transcriptions that included non-Roman alphabet characters.

Whisper did a great job at handling noise with 38/200 of the different native languages had no difference or reduced the WER. As for wav2vec, 0/200 had no difference or reduced WER. This is likely due to Whisper's massive amount of time spent on training data to be able to recognize words even with noise.

The lowest difference for Whisper was the speaker whose native language is Hainanese with -0.928 WER difference between clean and noisy data; for wav2vec, the speaker whose native language is German with 0.058 WER difference between clean and noisy data. The highest difference for Whisper was the speaker whose native language is Italian with 2.493 WER difference between clean and noisy data; for wav2vec, the speaker whose native language is Sa'a with 0.971 WER difference between clean and noisy data. While Whisper has a larger difference between types of data, it was on average the more accurate model.

Limitations and Future Work

Unfortunately, my computing power only allowed for the processing of 400 speech files across the 2 different ASR models to be finished by the deadline, the total time it took to process all of the data was 5 hours. Should someone choose to expand and continue on my work, they can use the 1,940 other speech files and mix it with the pure noise to use a total of 3,880 files for the expansion of the project.

In order to properly test the pure audio to compare across models, future researchers should go out and collect their own data instead of one that is open source because the current data could have been used to train the models, hence biasing the data.

Others can also try mixing different types of white noise, maybe one that sounded more human like in order to test for speaker identification and diarization. Another possible continuation of this project could also be looking at what causes the accented English speech to be transcribed as something else, looking at how far the pronunciation of the phonemes are from standard American English or British English. Researchers can also continue this project by looking into the different language families and analyzing if there are any similarities between the WER between families.

Conclusion

This study aimed to evaluate two ASR models, Whisper and wav2vec, in interpreting accented English under 2 different conditions, clean audio data and white noise mixed data. The experiment utilized 200 different speech samples with an additional 200 speech samples with noise totaling to 400 unique audio files.

My findings indicated that both Whisper and wav2vec have their own distinct strengths and weaknesses. While Whisper was trained on a significantly larger dataset, it demonstrated lower accuracy than wav2vec in clean audio transcription. It ended up making more complex transcriptions than wav2vec by adding the different characters and punctuation. However wav2vec had a tremendously difficult time transcribing the noisy audio in comparison to Whisper.

This experiment also identified language specific nuances in ASR performances where certain languages resulted in better transcription accuracy across both models.

I am excited to continue working on this experiment as I expand it to understand the different aspects and variables that come with conducting this project. I would like to touch more on the different language families and speaker information as seen in the results section with the Chamorro native speaker. There is a lot that goes into language acquisition and how people develop their accents which can affect the way they are understood by others and computers.

Although wav2vec does a better job at outputting English transcriptions with accented English, Whisper is the better model at transcribing audio. Even with noise added, Whisper was able to maintain a close WER to their clean audio transcriptions. It was able to

decipher the words even when the data was noisier. If Whisper only outputs English, I believe that it would end up having a lower WER than wav2vec.

Overall, this study touches on the intricacies between different ASR models, accented English, and environmental factors. As technology evolves, I can't wait to see how different ASR models become more accurate with all kinds of different languages and people of different backgrounds.

References

[1] Daar, Hassan
https://www.educative.io/answers/how-to-use-open-source-whisper-asr-in-python

[2] Facebook (2020) “Wav2vec 2.0: Learning the structure of speech from raw audio”
https://ai.meta.com/blog/wav2vec-20-learning-the-structure-of-speech-from-raw-audio/

[3] Marangon, Johni Douglas (2023) “How to calculate the Word Error Rate in Python”
https://medium.com/@johnidouglasmarangon/how-to-calculate-the-word-error-rate-in-python-ce0751a46052

[4] OpenAI (2022) “Introducing Whisper” https://openai.com/index/whisper/

[5] Styler, Will (2024) “Introduction to Automatic Speech Recognition”
https://wstyler.ucsd.edu/talks/l168_13_asrintro.html#

[6] Subramanian,Dhilip (2021)
https://www.kdnuggets.com/2021/03/speech-text-wav2vec.html

[7] Weinberger, S. (2013). Speech accent archive. George Mason University.