Name: Jason Zhou
Mentor: Dr. Dongjin Song
Status Report #: 9
Time Spent on Research This Week: 13.5
Cumulative Time Spent on Research: 73.5
Miles Traveled to/from Mentor This Week: 0
Cumulative Miles Traveled to/from Mentor: 0
==========

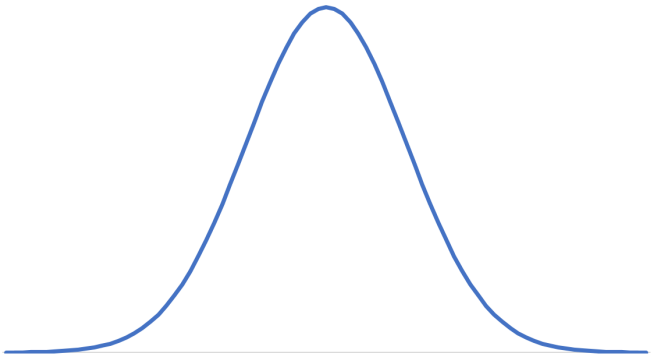
Monday, November 15th, 2021: (1 Hour)

I had my weekly zoom meeting with my mentor. Because the rough draft of the research proposal was due today, I asked my mentor to look it over. After reading it, he said that it was decent; however, he also said that I needed more detail about the methods I would be using and the justification for why I chose those methods.

Tuesday, November 16th, 2021: (3 Hours)

On this day, I began researching variational autoencoders (VAE). I wanted to look into VAEs because they could a good approach to use in my research project.

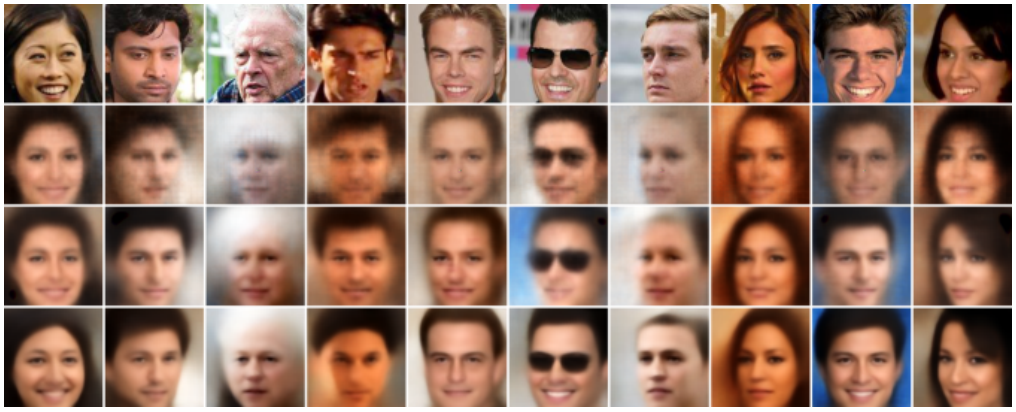
VAEs are what are called generative models, which means that they synthesize new data like images or audio files based on previous inputs. For regular autoencoders, the variables learned are derived from the inputs it receives. These variables are always discrete(meaning one value); however, for a VAE, instead of learning a discrete variable, it learns a distribution, or a range of possible values, for a certain variable. More often than not, this distribution will appear as a Gaussian.



(Example of a Gaussian distribution, which is also called a bell curve or normal distribution)

To generate new, unique images, the VAE samples a point from each Gaussian that is created from each variable and combines these sampled values to create a new image. With this being said, oftentimes the data that it generates will be similar to the original input. For example, if an image with a person with red hair was inputted, the VAE might output the same person but with slightly lighter colored hair. Additionally, although VAEs can be good for

generating new data, it is worth noting that these data can be of low quality.



(Picture that displays the images created by a VAE. The first row represents the images that were inputted into the model. The later rows represent the output of the VAE. As one can see, the later images are blurry and can be said to be of lower quality than the original inputs. Additionally, notice how the faces of each reproduction are slightly different compared to the original. This is one of the key features of a VAE)

Wednesday, November 17th, 2021: (3 Hours)

I spent this time looking through more research articles for the methods I would be using for my research. Specifically, I wanted to find low-level[1] pros and cons; however, I found this task hard to accomplish. Unfortunately, I spent this three hour period unable to find anything of importance. Many of the articles were hard to understand and used terminology I was unfamiliar with. In the future, I will need to find articles and resources that will help me break down the complexity of these articles.

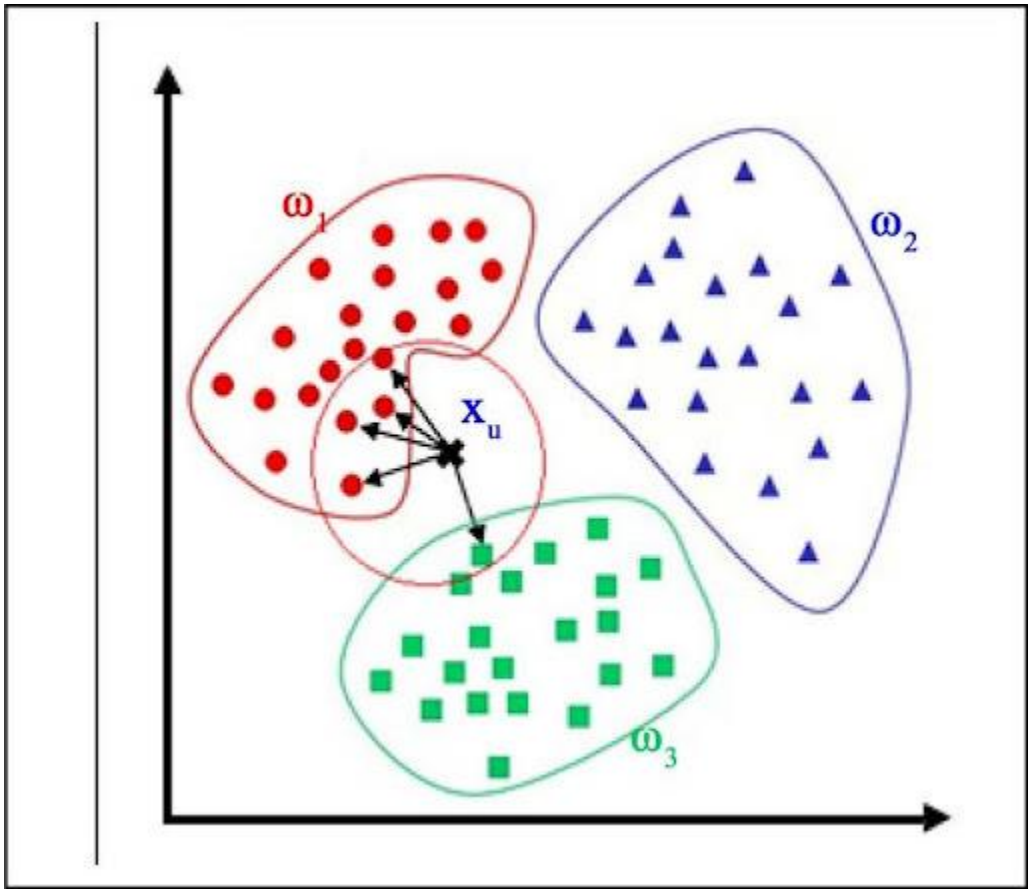
Thursday, November 18th, 2021: (3 Hours)

On the previous day, I was unable to find any information of use; however, while searching on the internet, I was able to locate some useful articles on the topics I was researching.

The article contained information on classic anomaly detection (ones that do not use deep learning). Specifically, these methods work purely off of clustering data and creating classifications for each group. Additionally, the article also explained each method in low-level and high-level detail, which made it easy to understand. In general, I was able to learn about three models that I was interested in: K-Nearest Neighbors (KNN) and Gaussian Mixture Models (GMM).

[1] Low-level refers to the specific mathematical operations that are happening inside the algorithm. Generally, low-level ideas are things that only experienced people within the field will understand. Conversely, high-level would be a general description, something anyone could understand

To start, KNN is a clustering techniques that focuses on a specific datapoint and scans for the closest datapoints within a certain radius. These surrounding datapoints are usually called neighbors (hence the name K-Nearest Neighbors). This process is usually done for every single datapoint (the radius that is scanned is dependent on the nth closest neighbor, thus the scanning radius can be different for each point). Points that fall within each other's vicinity can be designated as neighbors in a neighborhood. In the context of anomaly detection, any point that does not fall within a neighborhood (in other words, an outlier) is most likely an anomaly. Although this is usually a reliable way to detect anomalies, one can imagine the time it takes to complete this algorithm. First, the computer must calculate a certain radius to scan for a point and repeat this process for every point in the dataset. Thus, the larger the dataset, the more time and resources the algorithm will take. For the datasets that I am using, which contain thousands of pieces of data, this method would unfortunately not be able to find an anomaly within a timely fashion.



(An example of KNN that shows a picture of a graph that plots some data points. The different colored shapes represent a neighborhood. In the context of the picture, a new point "Xu" is being placed to see which neighborhood it belongs to by calculating its nearest neighbors. The groups will change and adapt based on new datapoints classification)

Next, I learned the pros and cons of Gaussian mixture models (GMM). These models work to cluster data in order to create probability distributions. For example, similar to KNN, GMMs work to self-create groups and classifications. Each group is converted into a probability

distribution. Simply put, a GMM tells the user the percentage, or chance, that a certain data point falls within a certain category (KNNs do not do this). The pros of GMMs are that the data can be clustered fairly flexibly and the process of classifying and creating distributions is also efficient. This is important because it provides an alternative clustering method compared to KNN. Specifically, using a GMM is more time and resource-efficient with larger datasets, making it the better choice for the dataset I was given by my mentor.

Afterward, I attempted to create a gaussian mixture model with the help of a YouTube video. Ultimately, what I learned is that each distribution in a GMM must be given a mean and a variance (standard deviation). These two attributes determine the shape of the curve. Additionally, I also learned that in GMMs, an EM (expectation-maximization) algorithm can be used to optimize the mean and variance as well as other hyperparameters used for making a GMM. Unfortunately, I did not finish making the GMM, so I plan to continue at a later time.

Friday, November 19th, 2021: (2.5 Hours)

After learning about GMMs, I decided to learn about log mel spectrograms. Although I investigated this topic in previous weeks, I wanted to learn how to actually extract these kinds of spectrograms from audio data because this would be a significant part of developing the models for my research proposal (the data must be converted into a manageable form for the models). First, in order to create a mel spectrogram, one must create a spectrogram and a mel filter bank. The spectrogram is simply made from converting the raw file into its visual representation, and the mel filter banks can be made by running a one-line command from a coding library called Librosa.

```
#mel filter bank
#apparently you matrix multiply this with a vanilla spectrogram and the result is the mel spectrogram

#n_fft = the frame size
#sr = sampling rate
#n_mels = number of mel bins
filter_banks = librosa.filters.mel(n_fft=2048, sr=22050, n_mels=10)
```

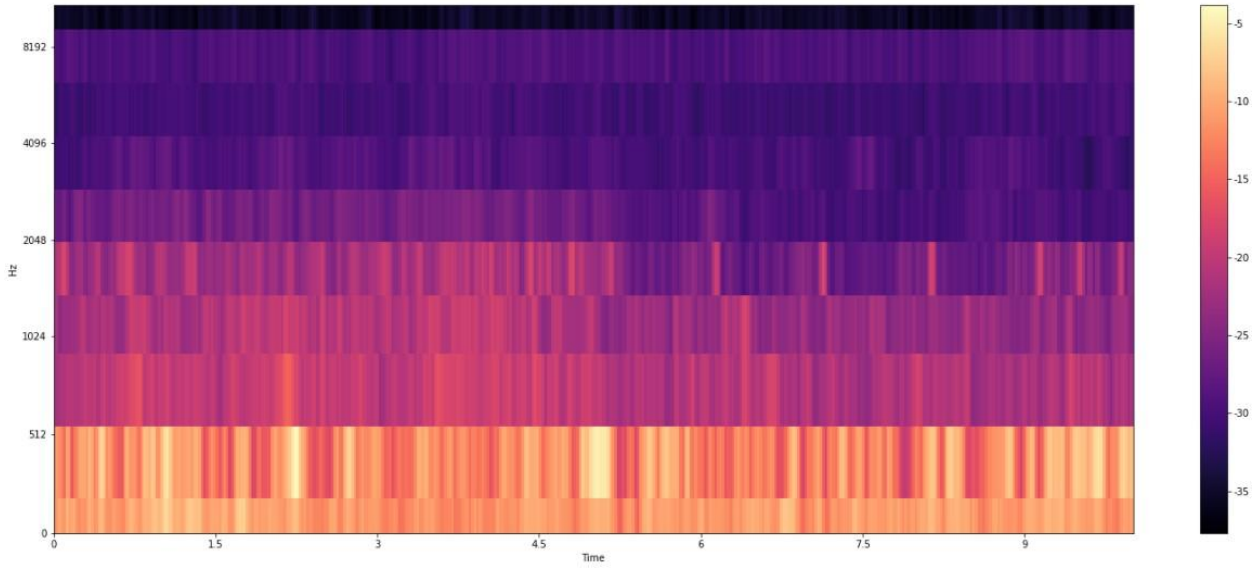
(A picture of my code for creating mel filter banks. The green text represents comments and does not contribute to the output of the code. Without the Librosa library, this would have a much, much more complex process.)

As it turns out, with the Librosa coding library, I do not need to create the regular spectrogram and only need the original audio file. As such, I can input the filter banks and raw audio files into a single line of code to make the mel spectrogram.

```
#extracting Mel Spectrogram
#y = signal/audio data (in this case 'af' is the input)
#sr = sample rate
#n_fft = frame size
#hop_length = number of samples between successive frames (the number of samples that overlap two frames)
#n_mels = # of mel bins
mel_spectrogram = librosa.feature.melspectrogram(scale, sr=sr, n_fft=2048, hop_length=512, n_mels=10)
```

(A picture of my code that makes the mel spectrogram. I am incredibly thankful that Librosa exists. This library has greatly simplified the process and has done most of the heavy lifting when creating the mel spectrogram)

For the final result, here is a picture of a mel spectrogram I extracted from an audio file taken from the dataset my mentor gave me.



(A picture of a mel spectrogram. This is actually a log mel spectrogram; however, I call it a mel spectrogram to simplify things. For the purposes of demonstrating the visualization of sound, they are essentially equivalent)

Saturday, November 20th, 2021: (1 Hour)

Besides extracting mel spectrograms as features of audio, there also exists a feature called the Mel-Frequency Cepstrum Coefficient, or MFCC for short. On this day, I watched a video on MFCCs.

The video I watched explained that audio files can basically be split up into two components: the spectral envelope and the glottal pulse. The spectral envelope contains the frequencies that make up the audio, while the glottal pulse contains the pitch. With MFCCs, people are mostly concerned with the spectral envelope, because this more or less contains the identity of the audio. Theoretically, to create an MFCC, the waveform must have a discrete Fourier transform applied to it and then be converted into a log-amplitude spectrum. Afterward, it is scaled using the Mel scale and a discrete cosine transform is used to calculate the MFCC. However, after watching the video, there were still some questions I had. Specifically, although MFCC utilizes the spectral envelope, what exactly is it measuring and outputting? Additionally, what are the coefficients that are mentioned in the name of MFCC, and what do they do? These are questions I will have to explore later.

References

Deep learning for anomaly detection. (2020, February). Cloudera Fast Forward. Retrieved November 22, 2021, from https://ff12.fastforwardlabs.com/

Raval, S. (2017, August 2). *Gaussian mixture models - the math of intelligence (week 7)* [Video]. YouTube. https://www.youtube.com/watch?v=JNlEIEwe-Cg

Velardo, V. (2020, September 17). *Extracting mel spectrograms with python* [Video]. YouTube. https://www.youtube.com/watch?v=TdnVE5m3o_0

Velardo, V. (2020, October 5). *Mel-frequency cepstral coefficients explained easily* [Video]. YouTube. https://www.youtube.com/watch?v=4_SH2nfbQZ8&t=2683s

Zhang, J. (2013). Advancements of outlier detection: A survey. *ICST Transactions on Scalable Information Systems*, *13*(1), 1-26.