Emotion Recognition with Exploratory Video Analytics System (EVA)

Seong Wook Choi*
schoi330@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

Bryan Kim
bkim441@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

Yu Yu
yyu@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia

ABSTRACT

Machine Learning is one the most rapidly evolving technologies, and researchers are continuously attempting to duplicate many of the functionalities that only the most complicated system to exist - the human brain - can perform. One specific area that drew the team's interest is the utilization of machine learning models to detect emotions, which is often a difficult task even for a human. While much work on detecting emotion exists, the team proposes including such features in the Exploratory Video Analytics System (EVA) for better usability. The system will return the timestamps of the detected emotions and have SQL-like commands, which will allow the user to query specific emotions with ease. This system has a considerable potential to improve the industries dependent on the customers' reactions, such as the music and service industries.

KEYWORDS

face detection, emotion detection, OpenCV, deep CNN, FaceNet, SQL, EVA

1 GITHUB LINK

https://github.com/seongwook8/eva_face_emotion

2 INTRODUCTION

Face recognition is a promising field of research due to its many practical usages. While some researchers emphasize face-matching, a field directed towards improving security and identifying specific individuals, this study focuses on detecting human faces in general from videos (compilation of images) and identifying the emotions of the detected faces.

There are many potential applications of face/emotion recognition systems. For a DJ responsible for turning on the right music at the right time, this system will allow the DJ to understand better the kinds of music his audiences enjoy based on their expressions. For a service industry manager, such as a hotel manager, this system will inform the manager if their service representatives satisfy the customers and also be used to educate new employees. For androids and robots that interact with humans, this system can aid the robots to keep track of actions that displease humans and help them avoid those actions in the future. In an era dominated by video information (i.e., Youtube, Netflix, Surveillance cameras), a proper querying system of visual data will lead to faster access to necessary information, further accelerating the development of machine learning technologies which often require massive training data.

This paper proposes incorporating existing face/emotion recognition models into the Exploratory Video Analytics System (EVA), with many benefits. First, users would no longer need to set up environments and download separate libraries for various recognition models. The EVA already includes object recognition and tracking models. With the inclusion of face/emotion recognition models, users would be able to run all of these machine learning models through simple SQL-like commands using the EVA instead of installing many different libraries. Setting up and working with multiple libraries can often take a significant amount of time during installation and troubleshooting due to dependency conflicts. However, all these functionalities would work seamlessly once the models are correctly incorporated into the EVA.

Another benefit of including face/emotion recognition models in EVA is lowering the learning curve to utilize the models. When used individually, users would need to read the documentation on how to use the models, and if proper documentation does not exist, users would have to devote much time going through hundreds of lines of code to figure out how to achieve their goals. Even after figuring out how to use the models, the base code often only returns limited outputs, such as the annotated video. In this project, we develop simple SQL-like commands that return various information about the video, such as the timestamps, occurrences of a particular emotion, and the face counts, which lowers the learning curve to use the models and diversifies the information that the users can obtain.

While this system has many potential applications, detecting emotions from videos poses a significant challenge compared to doing so on a still image. Videos are often 24 ~ 30 fps, meaning that 1 second of the clip is composed of more than 24 still images. As the length of the video increases, detection of emotions will require tremendous computing power. Adopting face/emotion recognition models into the EVA also poses a challenge due to potential dependency conflicts. The challenges that the team has faced during development will be explained in a later section.

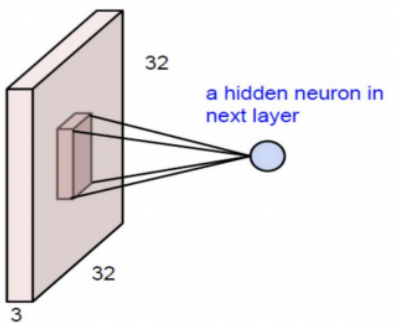

Figure 1: Illustration of how CNN works [1].

3 RELATED WORKS

3.1 Convolutional Neural Networks

Unlike the human brain, which perceives and processes images and videos as visual representations, modern computing devices perceive images as a collection of digital values representing each pixel, often in an RGB color model ranging from 0 to 255. Higher resolution images do not challenge the human brain in extracting information from the image. However, they pose a significant challenge to computers, with an increased number of pixels for processing. In today's world, full high-definition images have become the standard for images and videos, composed of 1920 x 1080 x 3 pixels per frame. Traditional Neural Networks would require 1920 x 1080 x 3 weights for a model with a single neuron, which can grow polynomially with every extra neuron. Even with only ten neurons, the model would require a total of 62,208,000 weights connecting every pixel to each of the neurons while most likely not providing useful information about the image due to the limited number of neurons. Even with major improvements in computing power over the last few decades, using traditional neural networks for image processing is still extremely slow.

Convolutional Neural Networks (CNN) was proposed to mitigate the issue of inefficiency in training neural networks with images that can easily contain more than a million inputs. Instead of using the entire pixels as an input, a filter or a kernel is applied to the image beforehand to reduce the dimension of the image. While there are many different types of filters, they all serve to reduce the dimensions of the image and maintain the information from the initial image as shown in figure 1. Another characteristic of the CNN is that it uses a bottom-up approach in detecting objects from images through multiple convolutional layers. For example, the first layer would detect edges from the image. The next layer could then use the information from its previous layer to detect shapes, which ultimately propagates to detecting entire objects in its final layers as illustrated in figure 2. Since face recognition and emotion recognition are also branches in the study of image recognition, they depend heavily on the concepts of CNN. Albawi et al. provide good explanation on the basics of CNN in their paper [1].

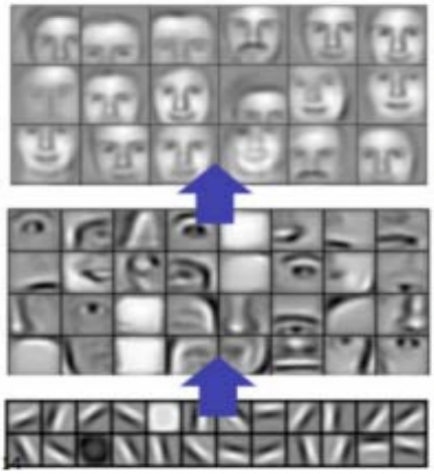

Figure 2: Bottom-up approach [1].

3.2 Face and Emotion detection

A convolutional neural network approach to face recognition was actually in the works as early as 1997 [6]. While video analytics and other advanced technologies did not exist at the time, image sampling and the fundamentals of CNNs were thought out and researched. The researchers' method of detecting faces was not too different from research studies today. The convolutional neural network was responsible for "extracting larger features in a hierarchical set of layers". Overall, by looking at key facial features and feeding a database full of faces to a trained recognizer, a group of researchers was able to build a model that can recognize faces in images, setting precedent for more advanced models to come.

While there have been steady advancements in the research regarding face recognition alongside the development of computing hardware, a noticeable improvement occurred when Schroff et al. developed a system called FaceNet [7]. FaceNet simplifies the task of face recognition to discerning a Euclidean distance to assess the similarity between the faces. A mapped model with this property achieves state-of-the-art performance, which records accuracy of 99.63% on the widely used Labeled Faces in the Wild (LFW) and 95.12% on YouTube Faces DB. These high accuracies are achieved while only using 128-bytes per face. Many existing face recognition libraries are based on FaceNet, including those that the team has explored for this project.

The training set on which the models are trained is also a crucial factor in determining its performance. Both Yi and Cao refrain from using the widely used LFW and created their own training dataset [3, 11]. Afterward, they validate the effectiveness of their created dataset, comparing a model trained on the commonly used dataset to a model trained on their own dataset. These articles exemplify the importance of diversity in the training face recognition model and provide a robust dataset that future data scientists can utilize for developing a better recognition model.

The Department of Computer Engineering at Kuwait University took on the challenge of developing a real-time facial expression analysis system in order to "classify physically disabled people and Autism children", which is an example of emotion detection used for medical purposes [5]. In this research, the students developed a model, using facial data from other university students, that would extract the facial features from an input image and send it through a CNN. The classification was done through the use of "markers" that mark the spot of specific landmarks on the face such as the eyes, the eyebrows, and the lips; these markers were then compared with the center of the human's face: the nose. The distance between these markers and the center point were what differentiated the emotions.

While this project is not mainly focused on real-time emotion detection, models like the one developed by Kuwait University provide value in asynchronous video analysis. Having a database that has a facial expression analysis feature will serve in many areas of Human Computer Interaction.

Another study that dives into the methodologies behind emotion detection was the one done by Dr. Prudhvi Raj Dachapally, who proposed two methods for emotion detection [4]. The intuition behind the first method of auto-encoders is to train a network for each emotion, which is interesting but more research needs to be done on improving the accuracy. The second method involved the convolutional neural network, with three convolutional layers, three pooling layers and two fully-connected layers. This method ended up with good results and the accuracy was high.

Arriaga et al. also use CNN to achieve real-time face detection, gender classification, and emotion classification by reducing the number of parameters required for training [2]. They even integrated their model into a robot called "Care-O-bot 3" to validate their results. The reported accuracy was 96% and 66% for gender and emotion classification, respectively.

CNN continues to be one of the most prominent models for face/emotion detection. As we will see in the later sections, many of our explored libraries leverage the effectiveness of CNN in their detections as well. Even with extensive work done on face/emotion detection not much work has been done to transform the video into database where SQL-like commands can be implemented to fetch practical information with ease. Therefore this paper proposes integrating existing face/emotion recognition libraries into the EVA.

4 RESOURCES

The team has access to only one machine with a dedicated GPU, which has the following specifications: 6 - core 3.6 GHz CPU, Nvidia GTX 1660 Super, and 16GB of DDR4 RAM. Without a dedicated GPU, testing the model during development will require a significant amount of time. Therefore the sample videos for testing during development will be of minimal length (~2 seconds), and the machine with the dedicated GPU will be reserved for more extensive testing of the implemented user-defined functions.

Sample videos are drawn from Pexels, which offers videos with no copyright for non-commercial use. The libraries that the team have explored include facenet-pytorch by timesler [8], Facial-Expression-Recognition.Pytorch by Wujie1010 [10], and Face-and-emotion-Recognition by vjgpt [9]. All of the libraries were developed using python and share some common frameworks such as Pytorch, OpenCV, Keras, and Dlib. The library by timesler detects faces, and the library by Wujie1010 detects emotions. These two libraries are based on pytorch, thus was ultimately selected as the libraries to be put together and implemented to EVA. While the library by vjgpt detects both faces and emotions, the face detection accuracy is lower than that of timesler. While main focus of this project is to integrate detection models into the EVA, but the team also aims to explore and combine multiple existing libraries to increase the accuracy of the models.

5 OBJECTIVES

Our initial goal was to research and implement an advanced machine learning model that can accurately detect emotions based on the facial features present on videos. We were interested in how the computer builds CNNs (convolutional neural networks) to locate certain emotions illustrated by facial landmarks. If there is one thing that computers were thought of as being incapable of pulling off, it was detecting emotions within humans because emotion is one of the most complex parts of the human brain, and we were centered on implementing a CNN that can look at a face in a video and accurately state which emotion is being expressed. However, due to the focus of this project being the EVA video database system, we decided to stray away from training an emotion detection model and go towards integrating an already-existing model as a user-defined function in EVA.

As our vision shifted from machine learning to adding a new functionality to EVA, we modified our goals to match that new vision. Until further changes are discussed, our project idea will revolve around implementing user-defined

functions, which will allow EVA to detect facial expressions. Our 75% goal will be to have a user-defined function that focuses on getting EVA to recognize faces in the video, and our primary goal will be to implement the function that detects emotions in videos. Having a pre-trained model integrated into a user-defined function and making it compatible with SQL commands will be the ultimate goal of this project. With these challenges in mind, the team has decided 75%, 100%, and 125% goals for this project:

- 75% - Integrate facial recognition as an User Defined Functions (UDF) in EVA for locating faces in videos
- 100% - Using the facial recognition feature, implement an emotion-detection model as an UDF into EVA to allow for videos in the database to have emotion analytics.
- 125% - Integrate SQL-like commands to query the video database and improve the emotion recognition model for higher accuracy.

6 CHALLENGES

The team has faced several challenges during the course of this project. The first challenge involved setting up the environment for the EVA. Plainly following the installation guide on the EVA Github did not work for the team mainly due to the conflicts in the existing miniconda version and the newly downloaded miniconda from the EVA installation. Also, the team's overall lack of experience with Linux systems only worsened the entire setup process. Even after completing the setup, as mentioned previously, only one of the team members had access to a dedicated GPU, which rendered running the sample code troublesome for team members without access to a dedicated GPU.

Another issue arose when exploring existing face/emotion recognition libraries. In order to test the different libraries, they had to be installed first, and the installation of multiple libraries often led to conflicts in the dependencies. The team had to spend extra time figuring out how to install the libraries and their dependencies, and again the limited access to a dedicated GPU meant more time would be wasted while testing the models.

In addition, getting familiar with the face/emotion recognition libraries and the EVA posed another challenge. Since the team was not involved in developing any of the libraries, in preparation to integrate a face/emotion recognition library into the EVA, we had to spend time familiarizing ourselves with the contents of the libraries. The outputs from the models were often a complete annotation of the videos, but we had to know how annotations were being recorded, the inputs to the prediction models, the variable types of the inputs and outputs, and the dimensions of the variables. Without proper knowledge of the libraries, we would not be able to develop SQL-like commands to obtain helpful information from the videos, let alone integrate the recognition models into the EVA for simple detection.

7 RESULTS

We were able to complete the 75% and 100% goals for this project, which is to successfully integrate face recognition and emotion detection as UDFs for the EVA. The 125% goal is only partially completed, through implementation of basic python functions which has the potential of being transformed into SQL like commands to return practical information from the detection algorithms, and also through improvement of the performance of the face and emotion detection models through combining multiple existing libraries.

7.1 User Defined Function

The most important step in implementing different libraries into the EVA as UDFs is to correctly implement the *getPredictions* function. This function takes in each frame of a video as the input, and outputs a list of labels, boxes and scores that can be obtained from the frame. For this project, the labels are the detected emotions, boxes are the top left and bottom right positions of the detected faces, and the scores are the prediction accuracy of the emotions. The pseudocode for the UDF implementation is provided in algorithm 1. The preprocessing steps to match the input formats have been disregarded in the pseudocode. Please refer to the actual code available in github for exact implementation of the UDFs.

Algorithm 1: Face / Emotion detection as UDFs

```
Function get_predictions(frame)
    boxes = face_recognition(frame)
    faces = []
    emotions = []
    scores = []
    for box in boxes do
        x1, y1, x2, y2 = box
        face = frame[x1:x2, y1:y2]
        emotion, score = emotion_recognition(face)
        faces.append(box)
        emotions.append(emotions)
        scores.append(score)
    end
    output = {'labels': emotions,
    'scores': scores,
    'boxes': faces}
    return output
```

Figure 3: vjgpt (top) vs timesler and wujie1010 combined (bottom)

7.2 GPU acceleration with Pytorch

As mentioned in the previous sections, the team has initially sought to use the Face-and-emotion-Recognition by vjgpt [9] to implement the emotion detection as a UDF. This library had both face and emotion detection already combined into a single library, which meant that the team would not have to spend extra time in an attempt to combine two different models to achieve our goals. Two problems arose whilst working with this library. First, the performance of the model was visibly lacking compared to other libraries. As can be see in figure 3, the vjgpt library cannot capture all the faces in the video accurately, when other libraries yield much better results. Also, since vjgpt was developed using tensorlow and keras, there was difficulty getting GPU acceleration to work after implementation. The performance of image processing libraries are highly dependent on GPUs, therefore having GPU acceleration capabilities are crucial in implementing UDfs. The performance difference with and without GPU acceleration is shown in Table 1. The results were obtained using the completed UDFs for this project.

Therefore a combination of timesler [8] (face recognition) and wujue1010 [10] (emotion detection) libraries is used for the final implementation of the UDFs. Since both of the libraries are based on pytorch, no issue arose in combining the two libraries or implementing them as UDFs, and the GPU acceleration was functional as well. As it is shown in figure 3, this combination of two separate libraries yield much better performing model compared to the previous library.

Table 1: CPU vs GPU runtime Comparison

Example	CPU	GPU	Improvements
1	39.78s	11.16s	3.56
2	27.52s	7.79s	3.53
3	47.75s	13.97s	3.42
4	33.70s	10.84s	3.11
5	27.07s	8.69s	3.12
		Average	3.35

7.3 Useful functions

Although the team did not manage to complete the implementation of custom SQL commands that can be directly used in the EVA environment, few useful functions have been developed to exemplify the potential use cases of the face / emotion detection models. This is the list of functions that are currently functional with the returned results from the UDFs.

- Counting the number of faces in each of the frames and returning a hash map of frame number and the number of faces.
- Returning the number of faces at a specified time. This provides more intuitive way to query the video since most video systems are based on time, not frame ids.
- Counting the number of emotions in each of the frames. Can also be done on the entire length of the video.
- Counting the proportion of the emotions present in a video or a frame.

8 EVALUATION

As the main focus of this project is to correctly implement face recognition and emotion detection libraries into the EVA as UDFs, we have evaluated the robustness of our implementation through various methods. First, the validity of the implementation was verified using the existing jupyter library in the EVA github repository, by only modifying the UDF's name from object detection to face recognition or emotion detection. This has shown that our implementation of the two libraries work without any issues. The jupyter notebook examples for the implemented UDFs can also be found in the project github repository, under the tutorials folder.

Also, as shown in figures 4 and 5, the UDFs were tested with multiple videos with variable lengths and all returned

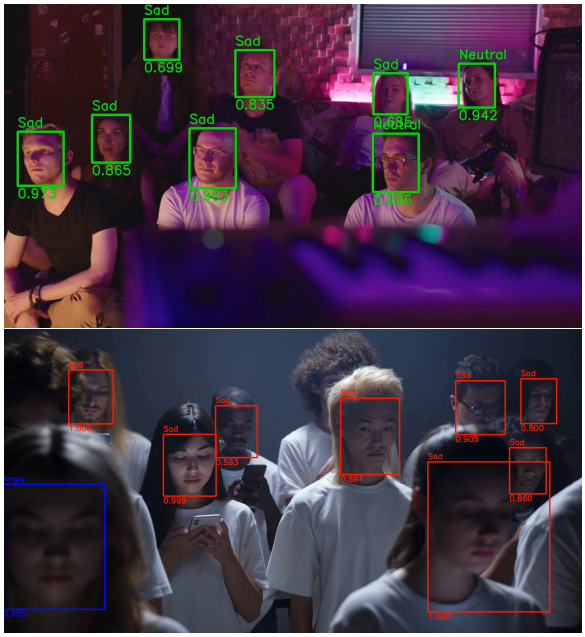

Figure 4

Figure 5

expected results. The bottom figure in figure 4 illustrates that our model can even handle cases where there are overlapping faces. All these results verify that our UDFs have been implemented correctly.

Since it is extremely difficult to assess the accuracy and performance of our implementations (Only relative accuracy can be tested with ease) the team has tested to see if the UDFs returned the same results with the libraries before implementing them as UDFs. Since the original libraries have accuracies reported on existing datasets, if the UDFs return identical results with the original libraries, we can make the claim that they have identical accuracies. Again, after testing on multiple sample videos, it was verified that the UDFs do in fact return identical results with the original libraries. The reported accuracies of the existing libraries are:

- Face Recognition: 99% on the CGGFace2 Dataset
- Emotion Detection: 73.11% on FER2013 Dataset

9 CONCLUSION / FUTURE WORKS

In this project, we have successfully achieved our goal of implementing face recognition and emotion detection libraries as User Defined functions in the EVA. These detection models will be useful especially to those who need to constantly monitor emotional feedback of a particular group of people, such as those associated with music or service industries. Now that the models have been incorporated into the EVA, potential users would reap the benefits of being able to use multiple video processing models with simple SQL like commands, without having go through the tedious process of installing multiple libraries and confronting dependency conflicts. The modified version of the face recognition and emotion detection model not only has high detection accuracies, but also has GPU acceleration capabilities, where the time spent on processing the video can be reduced drastically depending on the power of the GPUs that the users have. The robustness of the implementation has been verified with various samples, and therefore is expected to provide great user experience with minimum issues.

Some of the future work that needs to be done are transformation of the python functions into proper SQL commands so that users can directly access the results from the EVA. Also reducing the time required for processing the videos for recognition closer to real-time is another major point to be worked on for increased user experience.

REFERENCES

[1] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. 2017. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*. Ieee, 1–6.

[2] Octavio Arriaga, Matias Valdenegro-Toro, and Paul Plöger. 2017. Real-time convolutional neural networks for emotion and gender classification. *arXiv preprint arXiv:1710.07557* (2017).

[3] Qiong Cao, Li Shen, Weidi Xie, Omkar M Parkhi, and Andrew Zisserman. 2018. Vggface2: A dataset for recognising faces across pose and age. In *2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018)*. IEEE, 67–74.

[4] Prudhvi Raj Dachapally. 2017. Facial emotion detection using convolutional neural networks and representational autoencoder units.

arXiv preprint arXiv:1706.01509 (2017).

[5] Aya Hassouneh, AM Mutawa, and M Murugappan. 2020. Development of a real-time emotion recognition system using facial expressions and EEG based on machine learning and deep neural network methods. *Informatics in Medicine Unlocked* 20 (2020), 100372.

[6] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. 1997. Face recognition: A convolutional neural-network approach. *IEEE transactions on neural networks* 8, 1 (1997), 98–113.

[7] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 815–823.

[8] timesler. 2020. facenet-pytorch. https://github.com/timesler/facenet-pytorch.git.

[9] vjgpt. 2020. Face-and-Emotion-Recognition. https://github.com/vjgpt/Face-and-Emotion-Recognition.git.

[10] WuJie1010. 2020. Facial-Expression-Recognition.Pytorch. https://github.com/WuJie1010/Facial-Expression-Recognition.Pytorch.git.

[11] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. 2014. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923* (2014).