

# **Brad Pitt Face Detection Project**

Computer Vision - CIS 5810

Ryan French, Sukanya Joshi, Luisa Silva



We would like to thank our TA, Phillipe Do, and our Professor Jianbo Shi for all of their contributions in helping us gather the right knowledge and tools to successfully complete our final project in Computer Vision.

## **Introduction**

Imagine being able to recognize one of the most popular celebrities of our generation, Brad Pitt, in a video filled with different faces. We specifically chose Brad Pitt as our celebrity of interest because we are all huge fans of his movies, like Ocean's Eleven and Mr. and Mrs. Smith.

The main goal of our project is to be able to detect Brad Pitt's face amongst other individuals and track his movement throughout videos. For each recognized face in the video frame, we wanted to input it into our trained model and output a probability of it being Brad's. Then, we wanted to draw a bounding box around the face which has the highest probability of it being Brad Pitt.

We trained our model using a dataset filled with both videos and images so that our model could capture different angles of Brad Pitt's face. In order to build our model, we used a neural network that inputs features extracted from VGG-16 and outputs a probability of never seen faces being Brad Pitt or not Brad Pitt.

## **Objectives and Benchmarks**

Given a video filled with multiple people, we wanted to successfully detect and draw a bounding box around Brad Pitt's face along with the probability of how likely that face is Brad Pitt's. In order to achieve this challenging objective, we broke our project down into small benchmarks.

First, we tried to only test our model on a video which only consisted of Brad Pitt. Once we succeeded in this task and had a model with a high accuracy, we tested our model on a video with multiple faces including Brad Pitt's. Eventually, after finding different ways to improve our model with hyperparameter tuning, enhancing video quality of our test dataset, and speaking with our TA Phillippe Do, we were able to create the most optimal face detection model for Brad Pitt and our model was able to identify him in a video filled with him within a large group of individuals!

## **Tools Used**

We used Python for the entire project (i.e training the dataset, building the model, drawing the bounding box). As for building our training and testing dataset with Brad Pitt's face, we pulled various youtube and tiktok videos, as well as images from PubFig dataset<sup>1</sup>, compiled by Columbia University. Our pre-trained VGG16 model and linear network were created with TensorFlow Keras, and we specifically utilized the OpenCV package within Python to detect Brad Pitt's face.

## **Dataset and Methodology**

Our methodology is divided into 3 parts: building the dataset, training the VGG model with our dataset and applying the model to videos containing Brad Pitt.

To build the dataset we used online celebrity faces databases to extract 400 images of Brad Pitt and 900 images of other (non-Brad Pitt) randomly sampled celebrities. The other 500 images of Brad Pitt that completed our dataset were scraped from 5 different online videos; each of the 100 frames were randomly selected and added as images to our dataset.

---

<sup>1</sup> Link to dataset: <https://www.cs.columbia.edu/CAVE/databases/pubfig/explore/>

With our completed dataset, we started training our VGG-16 model. The reason that we chose to use a VGG-16 model was that it is a state-of-the-art pretrained model to detect faces. We passed all 1800 images through VGG-16, which outputted a vector of 25,088 floats per image (this is our input vector  $X$ ). Then we divided our data into a training set (900 images) and a testing set (900 images). For each input  $X$ , we chose  $y$  to be Brad (0) or Not Brad (1). Lastly we trained a 3 layer neural network that turned 2048 dimensions into 1: a probability of each image being Brad or Not Brad.

The last step was applying our trained model to videos where Brad Pitt is present. To do so we used the OpenCV Python package to iterate through every frame of a given video. For each frame, we used the Haar OpenCV function to recognize all the faces present in the frame. We then iterated through the coordinates of all those faces, so that we could input each of them individually into our model and get a probability out of it. We gathered all of those probabilities into a vector and created a bounding box, while displaying the probability the model outputs onto the face with the highest probability of being Brad. We then repeated that process for every frame in the video and saved the final product into a new .mp4 video.

### **Classification Model**

Our classification model utilizes transfer learning to output a probability that a given face is Brad Pitt's. Each of the images and frames in our training set, testing set, and live model is passed through a pretrained VGG-16 model. VGG-16 is a deep Convolutional Neural Network trained on a subset of the ImageNet dataset and is often used for feature extraction of face images. We imported the pretrained model from `tf.keras` and instantiated it without its three linear fully connected layers.

The input of this model is therefore a  $224 \times 224 \times 3$  image and the output is a  $7 \times 7 \times 512$  array of features. This output can be interpreted as 512 individual  $7 \times 7$  feature maps from the CNN. We then immediately flatten this array into a  $1 \times 25088$  vector and pass this through our own fully connected neural network.

Our network had the following structure:

Layer (activation)	Output Shape	Param #
input	(1, 25088)	
linear_0 (SELU)	(1, 1000)	25089000
linear_1 (SELU)	(1, 500)	500500
linear_2 (SELU)	(1, 50)	25050
linear_3 (SIGMOID)	(1, 1)	51

The last layer's sigmoid activation turns this network into a logistic regressor. Starting with an input of size 25,088, after 4 linear layers we are left with a  $Pr[Brad] \in [0, 1]$ . We use a threshold of 0.5 to classify our samples as being *Brad* (1) or *Not Brad* (0).

We trained this fully connected network on 1468 vectors of 25,088 features for 50 epochs and achieved a loss of 0.1055. We then tested this model on 367 vectors and yielded a Brad classification accuracy of 86.4%.

As you can see in the images below, our Brad Pitt face detection model predicted an actual image of Brad Pitt as having a 94.77% probability of being Brad Pitt, whereas an image of Cindy Crawford (non Brad Pitt) has a 8.07% probability of being Brad Pitt. This shows that our model proved to be successful.



Label: 1 (Brad Pitt)  
Model Output: 0.9477



Label: 0 (Not Brad Pitt)  
Model Output: 0.0807


## Face Detection



Once we had a successful face classification model, we began to classify frames of videos. Our algorithm is as follows:

- Get the next frame in the input video
- Use cv2.CascadeClassifier to identify all faces in this frame
- For each face:
  - Get its bounding box with cv2
  - Use this bounding box to crop the frame into a face
  - Run this face through our classification model to output a  $Pr[Brad]$
  - If  $Pr[Brad] > 0.5$ , add it to the list of candidate faces for this frame
- Declare the face with the maximum  $Pr[Brad] \in [0, 1]$  as the only Brad in the frame
  - If no face has  $Pr[Brad] > 0.5$ , output the image without any bounding boxes
- Draw a bounding box around this face and output the image


Success metrics for these output videos are based on the number of frames output with  $Pr[Brad] > 0.5$ . We would like to see the proportion of these frames be greater than 75%

## Results

Video	# Frames	Correctly Classified	% Correct	Video Screenshot
Just Brad 1	376	259	0.688	

Just Brad 2	325	307	0.944	
Brad in Friends	278	252	0.906	



Group at Table	270	237	0.877	
----------------	-----	-----	-------	--

### **Next Steps / Future Directions**

The next step in our project would be to extend our model to recognize more celebrities than just Brad Pitt. Therefore, we would have a multiclass classification in the end of our model and output the name of the celebrity that best matches the unseen data (image). That way, we would be able to display multiple bounding boxes on our output videos, depending on the amount of celebrities present in each frame. The first step in that direction would be to add more images to our dataframe (for each new celebrity we would want to recognize).

In addition, we would also want to improve our model's capability for recognition by adding images to the dataset where the celebrity (Brad Pitt, in this case) is seen from more diverse angles. Therefore, we would be able to better recognize faces when the celebrity is not looking straight to the camera (which presented to be a challenge in our output videos sometimes).

### **Challenges**

We faced numerous challenges throughout this face detection project. One of the first challenges we faced had to do with the format we were saving our output video (with the bounding box surrounding Brad Pitt's face). The function `VideoWriter_fourcc` from `cv2` package converts a string (four chars) to an int. For example, `cv2.VideoWriter_fourcc(*'MJPG')` gives an int for code MJPG and `cv2.VideoWriter_fourcc(*'mp4v')` gives an int for code mp4. That means that we have to specify the correct int depending on the format of our output video. Because we were specifying `*'MJPG'` inside the function but saving the video as a `mp4` format, our final output video was getting corrupted. However, we were able to quickly fix this with the correct code usage inside `VideoWriter_fourcc`.

Furthermore, another challenge we faced had to do with the usage of `resize` functions on images. At first we were using the `PIL` package `resize` function to `resize` our image before inputting it into

our model. However, that function was corrupting the output image so we decided to use Python's cv2.resize function instead. Cv2's resize method correctly made our image smaller so it could be pre-processed accordingly and inputted into the trained model..

Finally, given that most of the images we scraped of Brad Pitt for our dataset were of his face looking relatively towards the camera, our model had much better results on videos where he was mostly facing the camera as opposed to looking completely sideways. Our video images in our training set did have images with various side angles of Brad Pitt, however, for future directions, it would be good to add a larger amount of images with a more diverse set of angles of his face.

### **Related Works**

In this face detection article by Analytics Vidhya<sup>2</sup>, we found a method to use OpenCV to capture the video in a frame-by-frame manner, process each frame, and extract the locations of all faces in the image. This article guided us in our mission to detect Brad Pitt in videos using OpenCV. Furthermore, we found another article by Analytics Vidhya<sup>3</sup> which detects faces on recorded videos and has the ability to work on both windows and mac platforms, which we also used to guide us in our project as our team had different types of laptops (windows and mac).

In addition, we found a published article from Institute of Electrical and Electronics Engineers (IEEE) called Human Face Detection and Recognition in Videos<sup>4</sup>, where they used a technique called background subtraction to detect human faces as well as delved deeper into other methods like haar detection or eigenface method.

### **Conclusion and Takeaways**

Ultimately, with lots of trial and error, we were able to surpass our benchmark by building a highly accurate face detection model for Brad Pitt. We took a step-by-step approach, which helped us to complete this challenging project. From building the dataset from scratch to exploring various different packages to implement our model, we did thorough research on what would work the best for our face detection model and as a result, we learned a lot about the challenges of face recognition. We hope to continue building upon our project by including more individuals besides Brad Pitt in our dataset, such that our model can detect multiple celebrities, and also better detect side angles of our person of interest.

### **Acknowledgements**

We would like to give a huge shoutout to our TA, Phillipe Do, and Professor Shi for giving us the tools and knowledge to complete this project. We had weekly check-ins with Phillipe and he did an amazing job at helping us overcome various challenges listed above and informing us of different tools that we could use for our project.

---

<sup>2</sup><https://www.analyticsvidhya.com/blog/2018/12/introduction-face-detection-video-deep-learning-python/>

<sup>3</sup><https://medium.com/analytics-vidhya/face-detection-on-recorded-videos-using-opencv-in-python-windows-and-macos-407635c699>

<sup>4</sup><https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7732378>