Sprint 1

Team Spectre

Kyaw Htet Paing Win
Frankie English
Matthew Berrios
Maddie Powers
Brandon Hoang
Steven Johnson

Preamble

Why this idea?

- Wanted something we were excited about
- Felt it filled a need/market that was unexplored
- Great opportunity to implement machine learning

Machine Learning Models

- Kernel Method: does not scale well with large data sets
- Design Tree's/Decision Forests: good, but not optimal for image classification
- Neural networks

Our choice

- Neural Network
- Specifically Convolutional Neural Network (CNN)

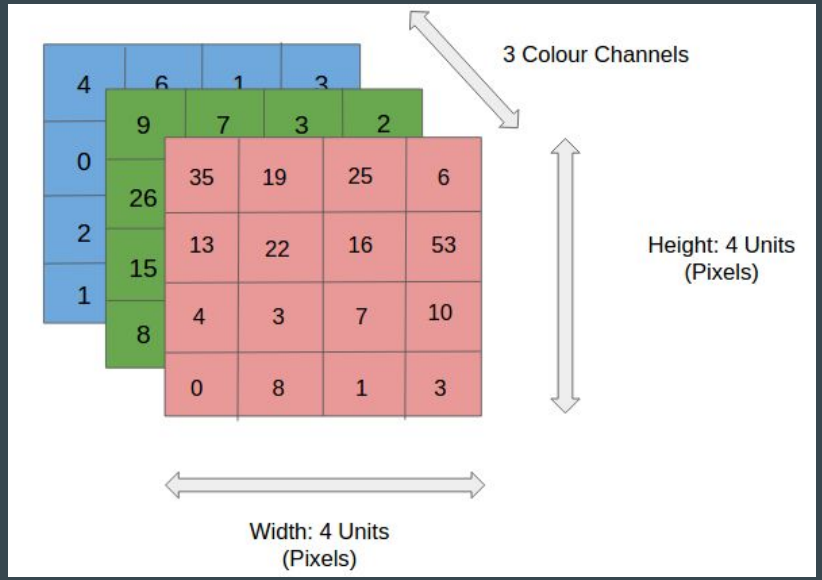
Computer Vision

- Computers cannot see, but what can they do for us?
  - Computer's can do large amounts of calculations quickly.
  - Neurons in human brains are good at recognizing patterns.
  - Loosely combining these and we have neural networks capable of "seeing".

- Slowly working our way up to more complex neural networks.
- Input is an image of a face, the output is a face shape.

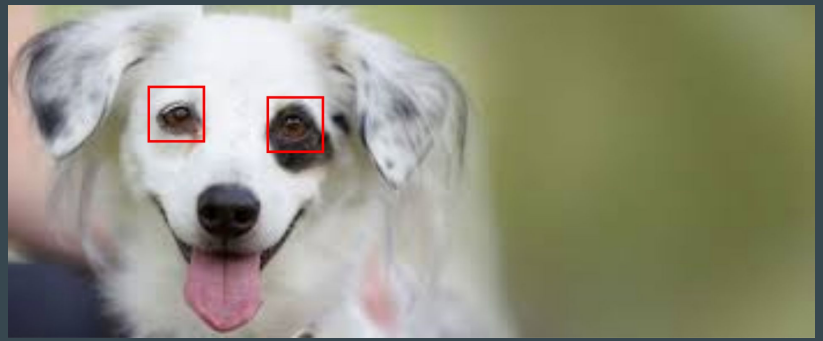
Convolutional Neural Network: Image Data

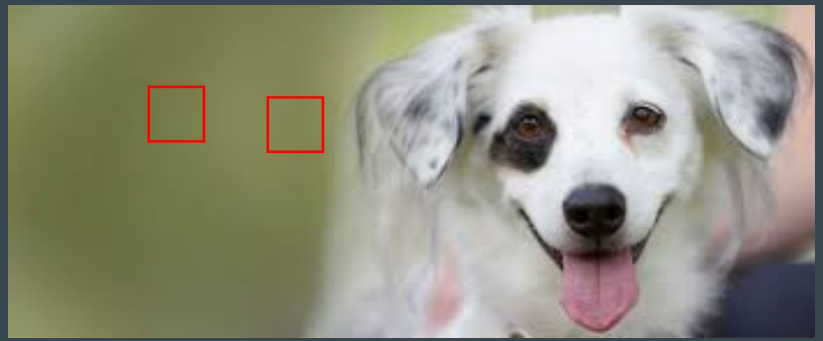
- Image data is made up of 3 dimensions: Image height, image width, and color channels
- The same image is represented by 3 specific layers: red, green, and blue.
- These channels tells us the color of each pixel



Dense Neural Network

- Dense Neural Network looks at the entire image at once and finds features.
- When it did find shapes and patterns it learned them in specific areas.
- Cannot learn local patterns and apply it everywhere like a Convolutional Neural Network





Convolutional Neural Network

- Each convolutional layer is going to examine a block of pixels in each image.
- Learns what something looks like and it can find that anywhere in the image.
- Each layer will scan through our entire image and pick up features and find features in the image and then based on the features that exist will pass that to a dense neural network and will look at the features found to determine specific objects.
- Compared to a DNN, CNN finds patterns that exists anywhere in the image because it knows what the pattern looks like not that it just exists in a specific area

Convolutional Neural Network: How it works

- The number of filters in a convolutional layer represents the depth of our response map
- If we are looking for 32 different patterns/filters than our output feature map will have a depth of 32.
- Each one of the 32 layers will be a matrix of some size containing values indicating if the filter was present at that location or not.

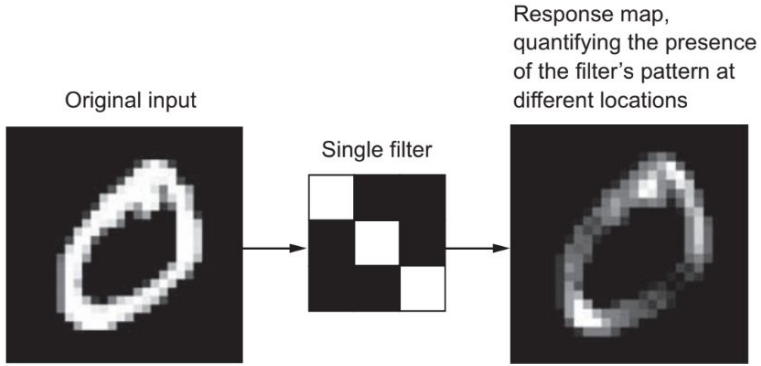
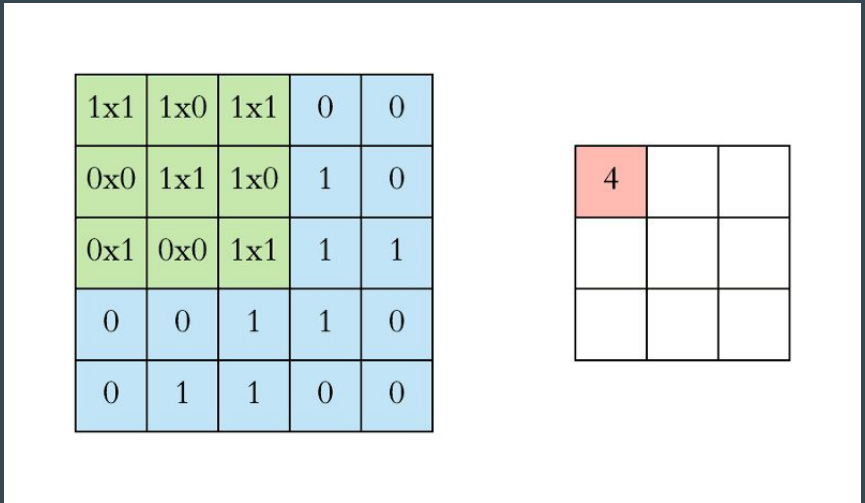
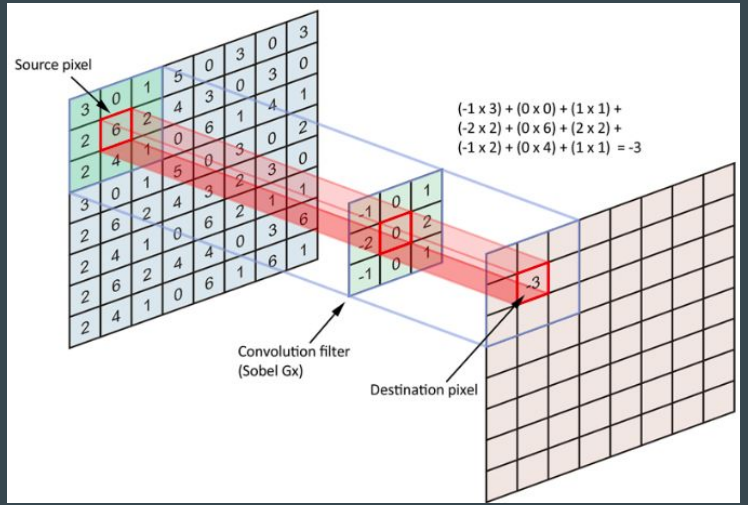


Figure 5.3 The concept of a *response map*: a 2D map of the presence of a pattern at different locations in an input

Convolutional Neural Network: How it works



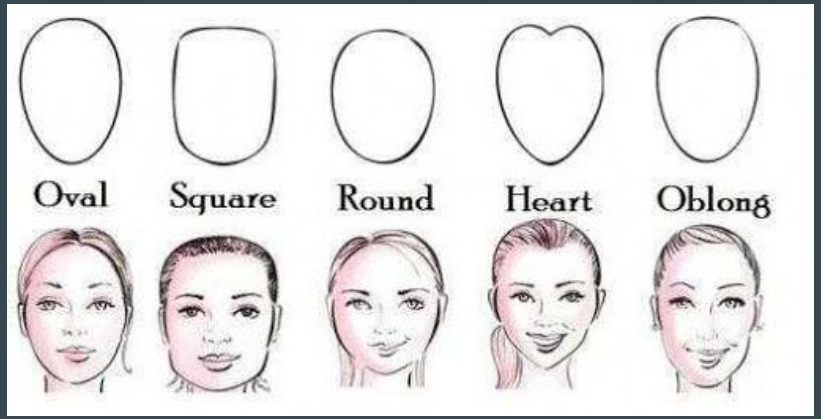
The filter slides over the input and the result is summed and added to the feature map.



The filter slides over the input and performs its output on the new layer.

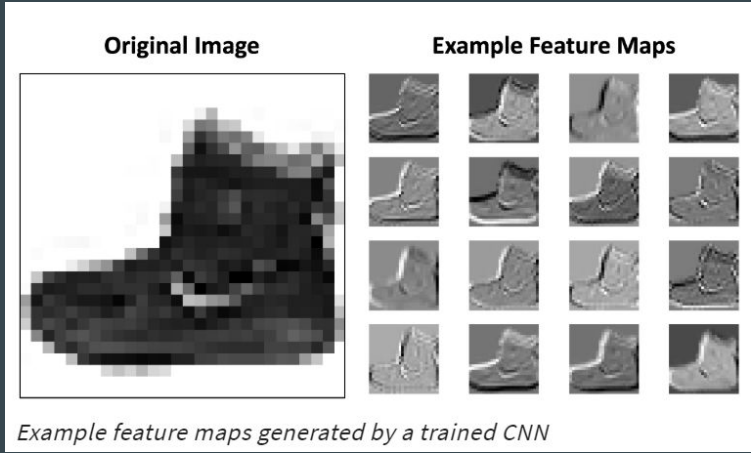
How it Applies to our Project

- Haircut for your face relies on face shape to come up with a recommendation for each user
- Being able to find the most accurate shape for a person's face is top priority
- The quality of our app will be directly affected by the accuracy of our model



Why we might want to use Convolutional Neural Networks

- Convolutional Neural Networks (CNN) are important as they can pull out features within an image in order to identify it.
- Take these shoes for example



Example feature maps generated by a trained CNN

Feature Extraction and Face Shape

- There are so many different types of skin colors, hair types, and facial hair/accesories that could be hard to account for when identifying a face.
- By essentially removing all of the extra unnecessary clutter in an image we can more easily identify the exact shape of a face.
- As faces can sometimes be only slightly different in actual shape, it's important to try and make clear distinctions as to what's round and what's square etc.
- We have identified 6 different face shapes with concrete definitions.

Resources

- Discriminative Deep Face Shape Model for Facial Point Detection (2014)
  - Proposes a facial point detection algorithm

(a) (b)

Fig. 1 Facial point detection. (a) Facial points that define the face shapes. (b) Facial images with detected facial poinits.

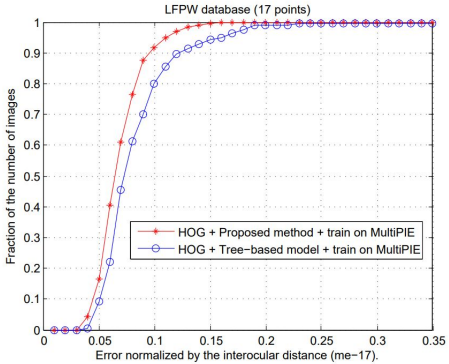


Fig. 9 Comparing the proposed discriminative deep face shape model with the tree-based model of FPLL method (Zhu and Ramanan, 2012) on the LFPW database.

Resources

- An Approach to Face Shape Classification for Hairstyle Recommendation, 2016
  - Algorithm predicted accurately 72% of the time

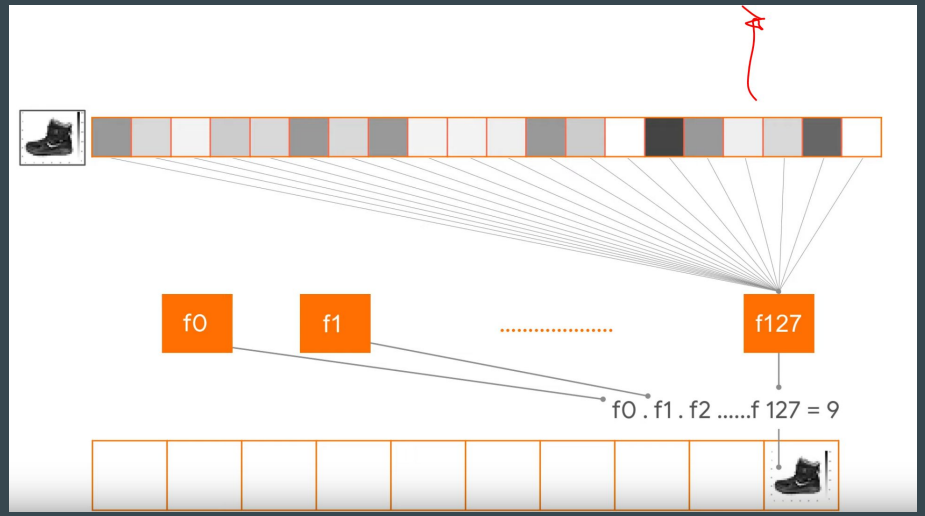
TABLE I: A SET OF PROPOSED DISCRIMINATIVE FEATURES FOR FACE-SHAPE CLASSIFICATION

Index	Description	Formula
1	Ratio of the height of a face to the width	$f_1 = \frac{d(p^{(9)}, p^{(18)})}{d(p^{(1)}, p^{(17)})}$
2	Ratio of the distance between left jaw and right jaw to the width of the face	$f_2 = \frac{d(p^{(5)}, p^{(13)})}{d(p^{(1)}, p^{(17)})}$
3	Ratio of the distance between chin and bottom of mouth to the distance between left jaw and right jaw	$f_3 = \frac{d(p^{(9)}, p^{(19)})}{d(p^{(5)}, p^{(13)})}$
4–11	Angle the x-axis makes with the straight line from a facial feature point to the chin point (the only facial-feature points considered start at right ear down to the chin)	$f_i = tan^{-1}(\frac{\lvert {p_y}^{(i-3)} - {p_y}^{(9)} \rvert}{\lvert {p_x}^{(i-3)} - {p_x}^{(9)} \rvert}); i = 4, ..., 11$
12–19	Angle the x-axis makes with the straight line from a facial-feature point to the chin point (the only facial-feature points considered start at left ear down to the chin)	$f_i = tan^{-1}(\frac{\lvert {p_y}^{(i-2)} - {p_y}^{(9)} \rvert}{\lvert {p_x}^{(i-2)} - {p_x}^{(9)} \rvert}); i = 12, ..., 19$

Resources

- Basic Computer Vision with ML (Tensorflow - YouTube)

```
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(128, activation=tf.nn.relu),
    keras.layers.Dense(10, activation=tf.nn.softmax)
])
```



Libraries

- OpenCV: Open source library of functions specialized for real-time computer vision (Java, C++, Python)
- TensorFlow: Data manipulation, dataflow for ML models
- Keras: Python neural network library, focus on being user-friendly
- Matplotlib: Plotting, graphs, etc. for Python and Numpy

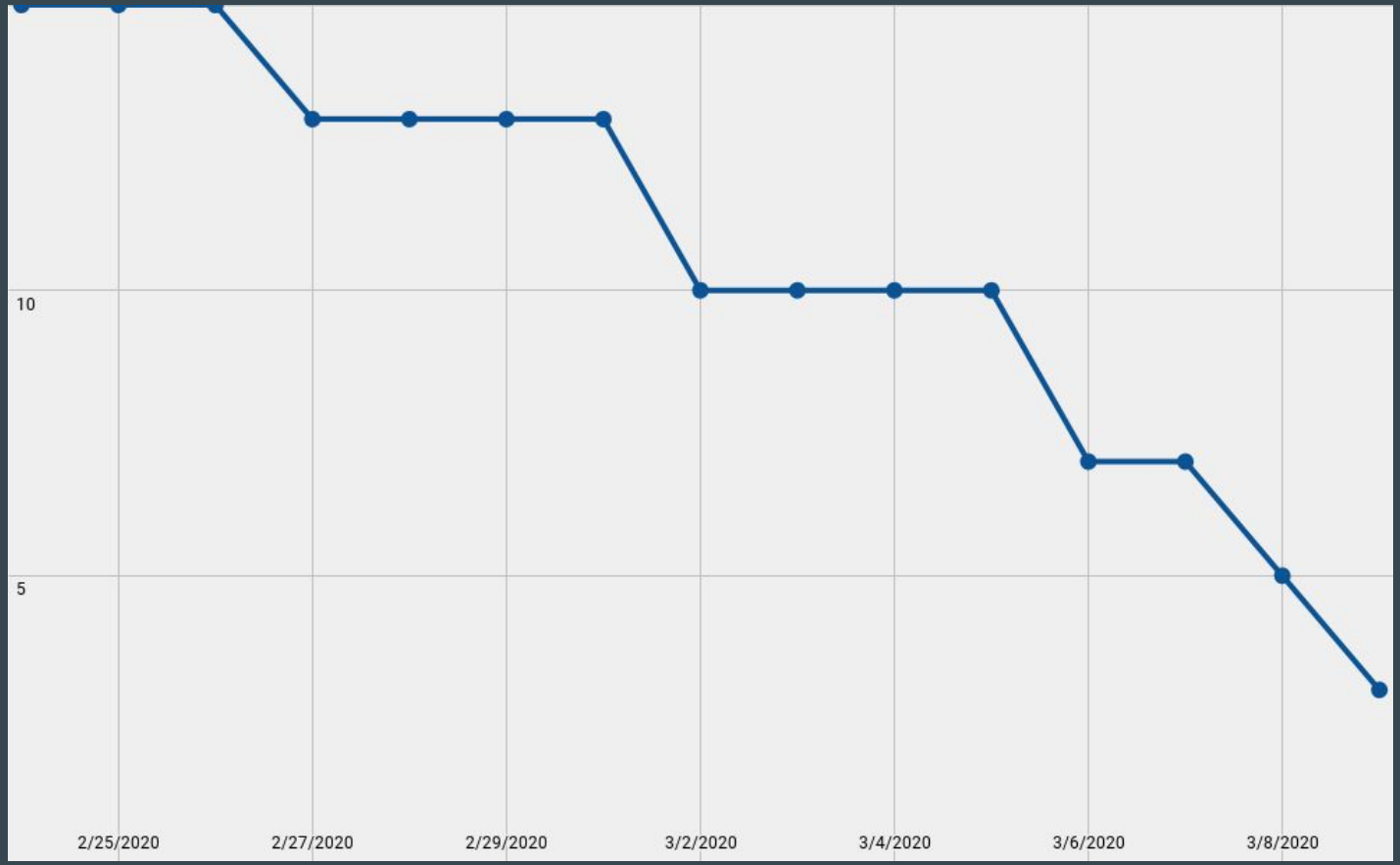
Data Accumulation

- Pre-existing data sets with face shape labels
- Reaching out to the researchers who have already done similar experiments
- Worst case:
  - Two people independently judge a subjects face shape
  - If they disagree, a third person will judge the face shape
  - Very slow but can fill in any holes we may have

Sprint #1 Result

- Sprint Goal: 90 Sprint Velocity and about 5/hr average/person
- Number of user story points planned: 15
- Number of user story points completed: 12

Burndown Chart



Sprint Review & Conclusion:

Held Sprint #1 review Monday March 9

Next 2 Models:

- Find a face within an image.
  - Draw a boundary around the face.
- Take a measurement of a face
  - Locate eyes on a person and measure the distance between them.

First Neural Network - Handwritten Digit Classification

7 Steps of Neural Network:

1. Load the MNIST dataset in Keras
2. Build the network
3. Preparing the network for training
4. Preparing the image data
5. Preparing the labels into category
6. Train the network on the training dataset
7. Evaluate the network on the training dataset