MACHINE LEARNING ENGINEER NANODEGREE

Capstone Project

Mariam Ashraf

August 31st, 2019

I. Definition

PROJECT OVERVIEW

This project a potential solution for Facial Expression Analysis as part of the Kaggle competition 'Challenges in Representation Learning: Facial Expression Recognition Challenge from ICML 2013'.[1] A paper was later published containing information regarding this challenge and two more in the Neural Information Processing journal.[2] The aim of the competition is to design a program capable of deciphering a person's feeling based on a photo of their face. Since human emotion greatly vary, combine, and are expressed in various ways, the project will simplify this by defining a few categories to aid the program in choosing from a small finite number of emotions instead.

The dataset is provided by the competition as 48x48 grayscale photos, some will be used to train the model, and others will be used to test how well the model performs on images it has not seen before.

This application can be helpful when implemented in robot-human interactions so the robot can have more information input to use when deciding on what to say or offer to the person speaking to them.

PROBLEM STATEMENT

The problem to be solved is to categorize a photo of a person's face into one of 7 emotions: Anger, Disgust, Fear, Happiness, Sadness, Surprise, and Neutral. The model should be able to make this decision for grayscale photos with a clear front shot of a person's face centered in the square photo. This problem should be solved using a Neural Network model. There are two potential solutions using a Neural Network, a Multilayer Perceptron (MLP) and a Convolutional Neural Network. It is more likely that a CNN will give better results since it accepts a 2D image as is, thus taking into account all the surrounding pixels and their position relative to one another, while an MLP will take a flattened image, where the importance of the relative position of the pixels is lost. However, an MLP has a simpler architecture making it less computationally taxing, so it is the preferred model for this project.

There is also a potential solution in Transfer Learning. This makes use of famously accurate image recognition models that were used for different purposes such as object recognition. After

adjusting minor aspects in the pre-trained model, it can be reused for our specific purpose. This saves some time and usually requires less computational power.

There are other potential models than the neural network which may be tested, but they are often too simplistic for classifying images so they will not be used as a main solution. Furthermore, the purpose of the project is to design a new model, thus, Transfer Learning will be a last priority.

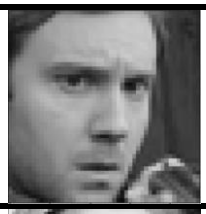
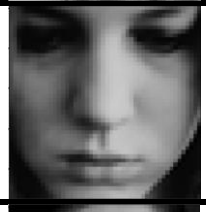
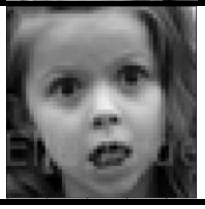
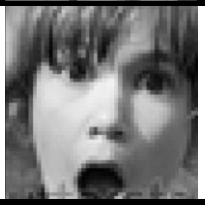
METRICS

As per the competition, the evaluation metric to be used will be accuracy, a simple percentage of how many times the model was correct over how many guesses were made overall. Since the dataset provided is labeled with the emotion displayed in the photo, it should be easy to check how many times the model was correct. This metric is the most accurate when the output required is in multiple category, not just a yes or no question, and when there are an almost equal number of training images in each category. The first criterion is met, and the second can be easily verified.

II. Analysis

DATA EXPLORATION AND EXPLORATORY VISUALIZATION

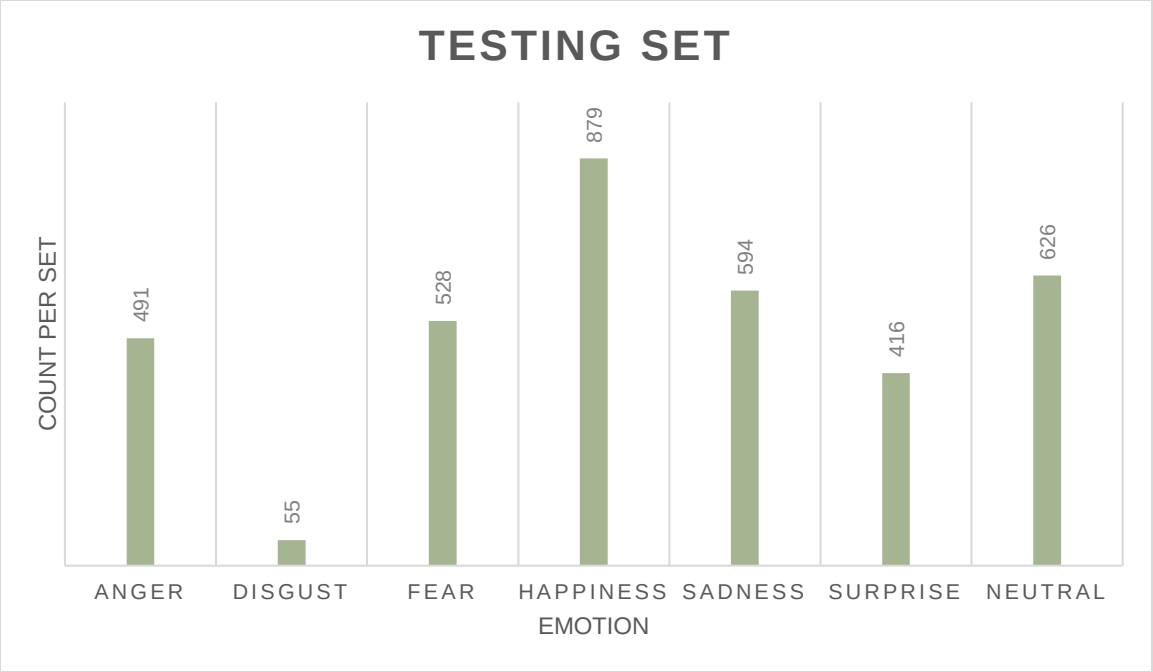
The dataset provided for the competition is made up of 35887 photos, each is a 48x48 grayscale image with a centered face as the examples below.

Anger			Disgust		
Fear			Happiness		
Sadness			Surprise		
Neutral					

The images are provided in an csv file with three columns. The first column is the label of the photo explaining the emotion in the photo which is signified by an integer between 0 and 6 inclusive. The integers mean the following: 0: Anger; 1: Disgust; 2: Fear; 3: Happiness, 4: Sadness; 5: Surprise, 6: Neutral. The second column is a flattened image: a series of 2304 numbers representing the color in each pixel. Since the images are grayscale, each number is between 0 and 255 depending on how light or dark the pixel is with 0 being white, and 255 being black. The third column signifies the usage of said image and thus splitting the set into three uses: 28709 images for training, 3586 images for public testing, and 3586 images for private testing.

At the time of the competition, the private testing set was not labeled, i.e. it did not have the emotion column, and the competitors were asked to provide the labels for judgement. The labels have since been made public after the competition ended. This provides us with two testing sets, one of which can be used for validation, to modify and improve the model during design, while the other can be used as the trust testing set to measure the accuracy of the model on new data.

Among the 7 emotions, the images are divided as follows:



It can be easily noticed that there is some skew in the data. For one, there are very few samples depicting Disgust and significantly more samples depicting Happiness than any of the other emotions. The rest range between four to five thousand photos in the training set, and 450 to 600 in the testing set. This may cause the model to have a maximum possible accuracy that is a bit too low since it will not be able to discern the important features in Disgust due to the small sample size, and may get too good at discerning Happiness at the expense of telling apart the rest of the emotions.

ALGORITHMS AND TECHNIQUES

As mentioned previously, the main potential solutions are either an MLP or a CNN. In the case of an MLP, the network will take as input the second column of the csv, that is the flattened images. There will be slight tweaking since upon reading the csv file, the array will be understood as a string, a sentence of characters, not as a series of integers. For easier handling as well, each image will be scaled so that the number representing the darkness of the pixel is between 0 and 1 instead of 0 and 255. In the case of a CNN, the input will need to be returned to the original square format as well as being scaled between 0 and 1.

There are several parameters to decide on such as the number of layers in the network or the number of nodes per layers as well as the batch size and number of epochs. The number of epochs can be kept at 100 and the improvements in accuracy monitored each epoch. One hundred epochs is usually enough for an MLP and most CNNs to converge to their highest accuracy, however, if a model continues to show frequent improvements in the last few epochs then the epochs will be

increased. The number of layers and nodes will have to be tested several times to determine the optimum number. The batch size will be set to 32 until the best model is reached, then different numbers will be tested to see if the model improves further.

BENCHMARK

There are three comparable accuracies that can be used as a benchmark for this project. First, the winning teams of the competition had accuracies between 65% and 71%. Second, the published paper stated that they tested humans at categorizing this set of images and found their average accuracy to be around 65%; note here that humans are particularly adept at this task. Third, also stated in the same paper, a null model garnered results of about 60%, and an ensemble of null models reached 65%. Based on those three models, a satisfactory result should be between 60% and 70%, with excellent results being over 65%.

III. Methodology

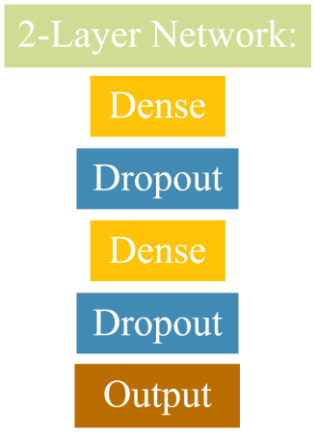
DATA PREPROCESSING

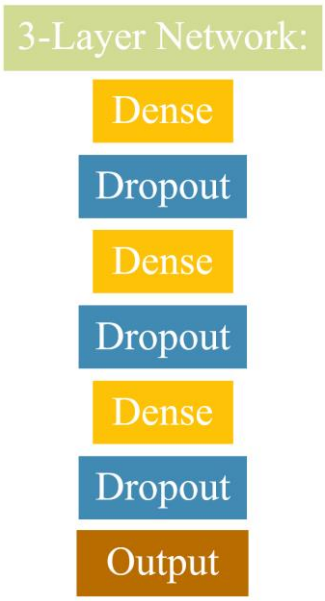
First, the data was read from the csv file and 3 arrays were made to separate the 3 columns. The data in the Pixels column was understood by the program as a string, that needed amending by transformation to integers so they can be understood by the model. The scaling was also done here. Then, the Usage column was used to separate the data into training, validation, and testing sets; and the labels were one-hot encoded. This means that instead of having one of the seven integers for a label, the results will have 7 columns where each row has a 1 in one of those columns signifying the emotion and 0s in the rest of the columns. This makes the data ready for input into a Neural Network.

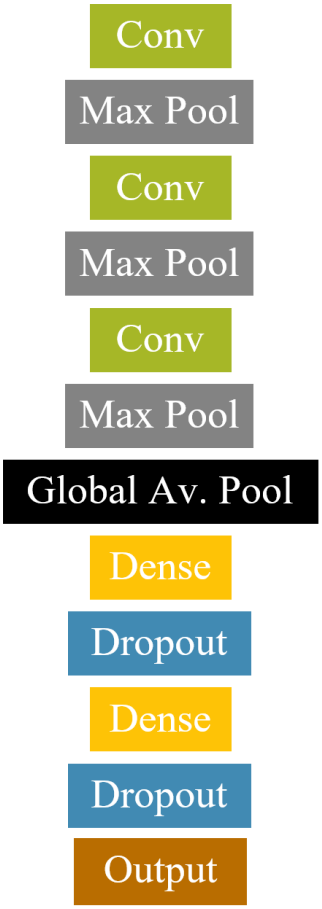
For input into a CNN, the data needed to be un-flattened. The process of un-flattening the data proved to be slightly more complicated than anticipated, yet very manageable. A new array was made and every picture that was un-flattened was appended to that array. An extra step was needed to edit the dimension of the array so the layers of the CNN can handle it. The labels were kept the same, one-hot encoded.

IMPLEMENTATION

The first trials were to achieve the results using an MLP since, as previously mentioned, it is simpler and faster to train. The main aim is to find an optimum architecture through testing different number of layers and nodes. Most testing was done over 2 or 3 layers with a variation in the number of nodes per layer as per the following architectures, where the dropout was kept at 10% in each layer:







As for the CNN, there was more variables to account for such as the filter size, stride, and pooling layers along with the number of nodes and layers and batch size. There were several different types of architectures tested, starting with only convolutional layers, then adding dense layers before the output dense layer, and several changes to the filter size and stride along the way. The trials made were far too many to summarize all in this paper, for example, it was quickly concluded that pooling layers were necessary for this classification problem, and so was a global pooling layer and some dense layers before the final output layer. Each of these additions increased accuracy by up to 15% converging to the architecture to the right, allowing for some potential tweaking for better results.

For both cases, the batch size was kept at 60 for testing, and tested at 10, 48, and 90 for the best model to check if there is potential improvement in changing the batch size. Furthermore, a checkpoint was made to save the best model based, the highest validation accuracy, across the epochs. The number of epochs was kept at 30 to decrease training time during testing, and was increased to 100 for the best model found to check if further convergence as possible.

REFINEMENT

Testing with MLP can be seen in the following table:

Run	Nodes/Layers			Training Accuracy	Testing Accuracy
1	48	96		31.27	32.60
2	48	96	192	37.47	36.97
3	48	96	192	39.10	38.09
4	192	96	48	44.15	40.15
5	192	96	48	41.09	40.17

Run 1 was used as a starter model. Several runs were made with increasing layers giving the 3-layer network of Run 2 as the highest accuracy. The next variable was the number of nodes per layer where increasing or decreasing the number of nodes in the 3 layers of run 2 proved to lower the accuracy of the model. Noting slight underfitting of the model so far, the dropout layers were removed giving run 3 with increased accuracy as predicted. Run 4 tested a slightly different architecture where the nodes decrease as the depth of the network increases and although the results increased slightly, there was an obvious overfitting. Accordingly, a dropout layer was added after the first layer. Testing different batch sizes for the optimum model concluded that the 60 was the best batch size. Retraining the best model for a 100 epochs did not improve the results and the best validation accuracy was reached in the 43rd epoch.

Overall, MLPs were always significantly less than the benchmark accuracy required and a CNN was essential to increase the accuracy. While testing, the following aspects proved to be essential to reach an acceptable accuracy.:

- Dense layers before the output
- Pooling layers between convolutional layers
- A global pooling layer
- Data Augmentation

Data Augmentation is a method of increasing the training set and its versatility by performing small changes to the dataset like shifting the images or mirroring them vertically or horizontally. The addition of data augmentation alone increased the accuracy by up to 10% and eliminated overfitting almost entirely in less complex architectures. From all testing the best accuracies reached were 64.1% and 64.8% using the following two models respectively:

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 32)	160
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 32)	0
conv2d_2 (Conv2D)	(None, 24, 24, 64)	8256
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 64)	0
conv2d_3 (Conv2D)	(None, 12, 12, 128)	32896
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 128)	0
conv2d_4 (Conv2D)	(None, 6, 6, 286)	146718
global_average_pooling2d_1 (	(None, 286)	0
dense_1 (Dense)	(None, 512)	146944
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 286)	146718
dropout_2 (Dropout)	(None, 286)	0
dense_3 (Dense)	(None, 7)	2009

Total params: 483,701.0
Trainable params: 483,701.0
Non-trainable params: 0.0

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 32)	160
conv2d_2 (Conv2D)	(None, 48, 48, 32)	4128
max_pooling2d_1 (MaxPooling2	(None, 24, 24, 32)	0
conv2d_3 (Conv2D)	(None, 24, 24, 64)	8256
conv2d_4 (Conv2D)	(None, 24, 24, 64)	16448
max_pooling2d_2 (MaxPooling2	(None, 12, 12, 64)	0
conv2d_5 (Conv2D)	(None, 12, 12, 128)	32896
conv2d_6 (Conv2D)	(None, 12, 12, 128)	65664
max_pooling2d_3 (MaxPooling2	(None, 6, 6, 128)	0
conv2d_7 (Conv2D)	(None, 6, 6, 286)	146718
conv2d_8 (Conv2D)	(None, 6, 6, 286)	327470
global_average_pooling2d_1 (	(None, 286)	0
dense_1 (Dense)	(None, 512)	146944
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 286)	146718
dropout_2 (Dropout)	(None, 286)	0
dense_3 (Dense)	(None, 7)	2009

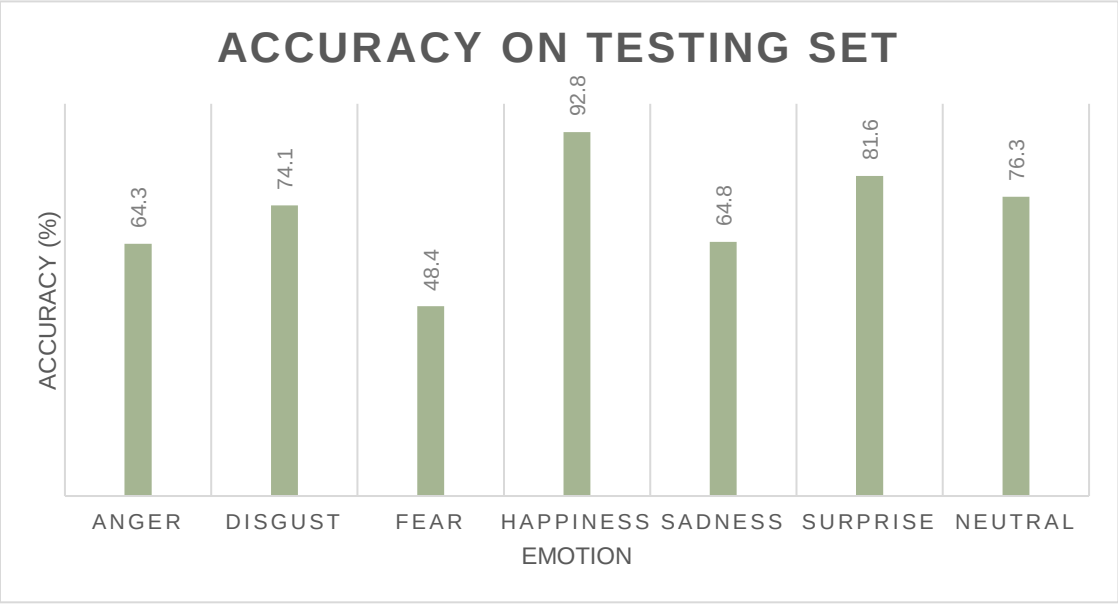
Total params: 897,411.0
Trainable params: 897,411.0
Non-trainable params: 0.0

Note that filter size and stride was kept at 2 and 1 respectively for all layers and dropout was kept at 10%. For both models, the training accuracy was at 71%, indicating some overfitting.

IV. Results

MODEL EVALUATION AND VALIDATION

A way of validating the model is looking deeper into its accuracy on each emotion. These results are shown in the following graph:



As anticipated, the model performs exceedingly well in “Happiness” since it has the largest number of images in the category “Happiness” by a fair margin. It was also anticipated that the lowest accuracy would be in “Disgust” since it has the least images, yet, its accuracy was not the lowest. This may be explained by the photos in the dataset. Some expression in the examples earlier are heavily exaggerated, like in Surprise, which would make them easier to spot as supported by the accuracy of the model in that emotion, while others are subtle such as the examples for Fear which the human author of this report would not have categorized correctly either.

Despite the generally good performance of the model, it is not very robust due to the varying accuracies that depend on which emotion is being tested. This may be improved by using a better dataset.

JUSTIFICATION

As mentioned earlier, the benchmark model is between 60% and 70% accuracy and an excellent model should reach more than 65%. Although the final model here was at 64.8% and thus in range of the benchmark, it still fell short of the excepted results of above 65%. These results are acceptable as first trials in solving the problem of facial expression analysis, yet, they cannot be used in real world situations until models can reach accuracies of more than 80% or even 90%.

V. Conclusion

REFLECTION

In summary, the aim of the project was to design a model that is able to distinguish between different emotions based on a person’s facial expression. The dataset provided focused on 7 emotions, used only grayscale images all sized to 48x48 pixels, and provided 3 sets of labeled data. To achieve this goal, two models where tested, MLPs and CNNs, where the former is simpler, takes less training time, and accept flattened images as input and the latter is more complex but is more likely to perform better because it can take a multi-dimensional input. According to several models, an adequate resulting accuracy should be between 60% and 70%.

The data was loaded and split into training, validation, and testing sets and the labels were one-hot encoded. First, the already flattened input was tested with different MLPs with varying results but the highest accuracy was more than 15% less than the benchmark required. CNNs performed better by continually reaching more than 60% accuracy, but none of the ones tested passed the 65% marker.

It is particularly interesting in this project to note the drastic improvement that occurred when a CNN was used instead of an MLP. It aids in showing how a flattened input loses a lot of important details that prove crucial for proper classification.

IMPROVEMENT

There are some aspects of the projects that can help in improving the results. As one may recall, the dataset was not equally distributed, giving the model very few images to train on to recognize Disgust, for example, and a drastically larger number of images in Happiness. So, a potential improvement is to use a better dataset that is more balanced. It can also be noted in some of the example photos that some of the faces are obscured by hands which may act as a red herring for the model. The dataset should be clear and have a view of the face only.

Secondly, an ensemble of CNNs can be used. This was a solution stumbled upon while testing but was difficult to implement and would have been more computationally expensive.

VI. References

[1] "Challenges in Representation Learning: Facial Expression Recognition Challenge | Kaggle", *Kaggle.com*, 2019. [Online]. Available: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/overview.

[2] I Goodfellow, D Erhan, PL Carrier, A Courville, M Mirza, B Hamner, W Cukierski, Y Tang, DH Lee, Y Zhou, C Ramaiah, F Feng, R Li, X Wang, D Athanasakis, J Shawe-Taylor, M Milakov, J Park, R Ionescu, M Popescu, C Grozea, J Bergstra, J Xie, L Romaszko, B Xu, Z Chuang, and Y. Bengio (2013). Challenges in Representation Learning: A Report on Three Machine Learning Contests. *Neural Information Processing*, pp.117-124.