Deep Learning Model for Human Emotion Analysis

Danqing Liu dqliu9@tamu.edu
Muhammad Hussain Anwaar mitu246@tamu.edu
Raghav Hari Krishna V S vsraghavhk@tamu.edu
Shashank Kamath Kalasa Mohandas shashankkamathkm@tamu.edu
Sungje Bang bang620@tamu.edu

Introduction

Many today experience negative emotions throughout the day. Studies show that negative emotions impair peoples ability to use logical reasoning. They also show that stronger negative emotions such as anger cause hypertension, which can become very deadly to people. To illustrate this example, Centers for Disease Control and Prevention state that in 2013, over 360,000 Americans died due to hypertension being the biggest factor (1,000 deaths each day!).

Emotion detection also improves the human-computer interaction in modern AI research. It can also benefit hospitals to study and help their patients better and to personalize the services provided by hotels to their customers.

In this project, we developed a project that can detect human emotions with the images of peoples facial expression and offer advice and recommendations that can help them quell their negative emotions if detected. We proposed a deep neural network ensemble model to predict facial emotions. Based on these predictions the system we designed suggests the user way to improve their mood.

The dataset we used is FER2013 human faces dataset with seven emotion labels. We have done experiments using VGGnet, Resnet50, mini-Xception[3], Inception-V3, Traditional CNN, Ensemble model. We done experiments on Google colab and HPRC terra cluster platforms respectively. The results shows that the ensemble model on with VGG19, Mini-Xception and Simple CNN can get 68.7% test accuracy.

Literature Review

There have been numerous developments in the Facial Emotion Recognition, since the boom in computer vision. Most of the times the data available for the Emotion recognition is unconstrained, where the conditions for same type of image are different. If we take the example of the FER 2013 dataset, images are not centred and has a high degree of variation among them. Before, training a convolutional model on these kinds of datasets, a certain degree of preprocessing is required.

Variations that are redundant to facial expressions, such as diverse backgrounds, poses and illuminations, are pretty much indigenous to unconstrained scenarios. Therefore, there are some methods and algorithms developed previously to pre-process the data in order to clean it for training purposes. The three fundamental components of image pre-processing are Face Alignment, Data Augmentation and Pose Normalization.

Face Alignment

It is a traditional step in the data cleaning and processing in the field of computer vision. Given the training data, we first detect the face, remove the background and non-face areas. The V&J (Viola and Jones) face detector is widely employed in the implementation of this principle. This method is fairly robust and computationally efficient for the near-frontal face detection.

Data Augmentation

Facial Recognition task are usually done using deep neural networks. These DNN require enormous amount of data for training purposes to avoid overfitting. Whereas, most of the times the data available for FER tasks is insufficient to train the model just on them without overfitting. Thus, data augmentation is way where we can increase the length of the data for better results and accuracy.

Pose Normalization

Another common yet intractable problem in dealing with unconstrained image dataset is the variation in poses among the images. Some studies incorporating pose normalization techniques yielded frontal facial views for emotion recognition. Numerous algorithms have been proposed for this task. One algorithm suggests that after localizing the facial landmarks, a generic reference model with 3D texture is being generated for all the faces. Which helps estimate visible facial components efficiently. Then we can synthesize the initial frontal face through back

projection. Where, each input image is back-projected to the reference coordinate system.

Deep learning is a technique which attempts to seize high-level abstractions via hierarchical structure of non-linear transformations. Where, there are different layers used to extract features out of an enormous image space using different convolutional operations.

Convolutional Neural Network

Convolutional Neural Networks has always been found robust to different representations of a facial image. They have proven their significance over the years by tackling images with varying scales and face-location changes. They have been better at meaningful feature extraction as compared to traditional multi-layer perceptrons. CNN can perform better at addressing problems of translation, rotation and scale invariance in the facial expression recognition.

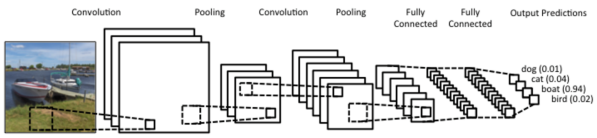

Figure 1. Convolutional Neural Network

A Convolutional Neural Network consists of convolutional layers, pooling layers, and Fully Connected Layers. The Convolutional layer has a set of filters to convolve through whole image space and extract meaningful features out of it. The convolution operation has several benefits such as local connectivity, which learns correlations among neighbouring pixels; weight sharing in the map which greatly reduces the number of parameters to be learned; and then the last but not the least is shift-invariance to the location of the object. The pooling layer that follows the convolutional layer, reduces the spatial-size of the feature map. Pooling is a non-linear down sampling technique that uses tools like average pooling and max pooling to reduce the dimension drastically.

Finally, there is a fully connected layer that is usually incorporated at the end of the network. This layer is just a multilayer perceptron layer where each layer is fully connected with each of the neuron in the proceeding layer.

VGG

VGG is a form of convolutional neural network model which has a deep complex architecture as compared to the traditional CNN. The VGG is being developed on the idea of how we can increase the depth of the network to learn non-linear features more efficiently, that too at a lower cost.

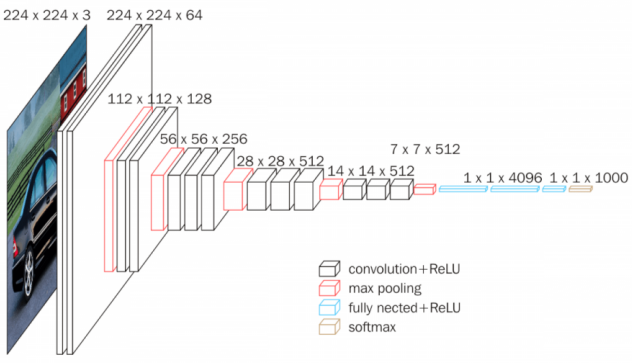

Figure 2. VGG Network

In some of the previously developed models, where people used larger sized kernels to learn the features from an image space to reduce the cost of computation. The VGG model proposed that the cost of using multiple smaller sized kernel is smaller than using a smaller number of larger sized kernels. Thus, there are multiple smaller sized kernels stacked on top of each other to learn features. These kernels increase depth of the network by stacking non-linear layers together. This enables the network to perform a better task and learn complex features with a cost which is small as compared to the other architectures.

Just to get our head around how this principle works, we can take a simple example of using three 3 by 3 kernels as compared to one 7 by 7 kernel. The parameters in the first case are $3*(9C^2)$ as compared to $49C^2$. The cost associated with multiple 3 by 3 kernels is significantly less. Also, 3 by 3 kernels help in retaining granular properties of an image.

The VGG has 13 convolutional layers are followed by 3 fully connected layers. The width of the network starts at a small value of 64 and increases by a factor of 2 after every sub-sampling/pooling layer.

GoogLeNet / Inception model

In a convolutional operation at a point in the network, each output channel is connected to every input channel, which is why it is called the dense connection network. However, most of the activations in the network are redundant. The GoogleNet builds on this idea that because of the correlations among them, only some of the activations are necessary. Therefore, the most efficient of the deep neural network architectures would have sparse connections among the activations. So, the output channels at one point would not be connected with all the input channels in the proceeding layers.

GoogleNet devised an Inception Module that approximate a sparse CNN with a normal construction which is not so dense. Due to a smaller number of neurons being active, the size of the kernels is kept small. So, this Inception Module in GoogleNet model uses kernels of varied scales such as 1*1, 3*3 and 5*5.

The Salient feature of this module is the 1*1 bottleneck layer, which reduces the cost of computation drastically. If we take the first Inception module of GoogleNet which has 192 input channels. It has 32 filters of size 5*5 and 128 filters of size 3*3. The order of computation is 25*32*192 for a 5*5 kernel, which can blow up the computations as we go deeper down the network. Thus, to avoid this, the module uses 1*1 convolution before applying 5*5 kernel. This has the total computation of 16*192 + 25*32*16. Thus, the computation using Inception module is reduced drastically.

In the figure below, we have depicted the very basic architecture of an inception module.

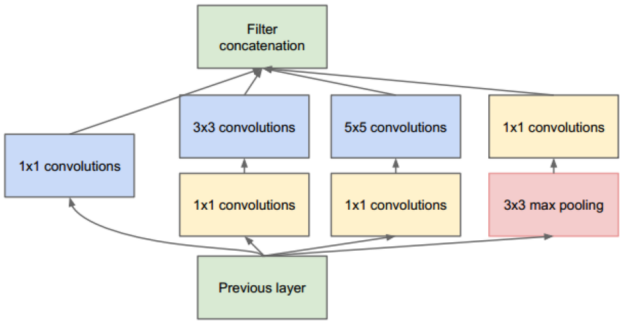

Figure 3. An Inception Module

The figure below is a representation of Inception V3 model. Here different number of inception modules are being used at subsequent steps to reduce the computational cost. The architecture is 22 layers deep but has a reduced number of parameters of 4 million.

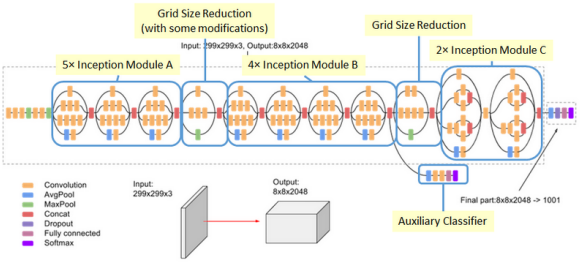

Figure 4. The Inception Model

Residual Network (ResNet)

So far in our discussion we have made up a point that increasing the depth of the network increases its accuracy, only if the overfitting is been taken care of. But there is a caveat, with the increased depth there arises the problem of vanishing gradient. A vanishing gradient is the problem when the signal required to change the weights of the layers at the start of the network becomes so small due to the large depth of network. Which is why the weights dont get updated and the problem arises.

There is another problem with training deeper networks. While performing optimization on huge parameter space and adding layers lead to a high training error. This phenomenon is called the degradation problem.

Resnet proposes a fine solution to cater these problems. It says if we use a residual block that creates a direct path between Input and Output, as it implies an identity mapping in the block and then add a layer just to learn the features on top of the indigenous input. As, this added layer only learns the residual, the whole block is called a residual module.

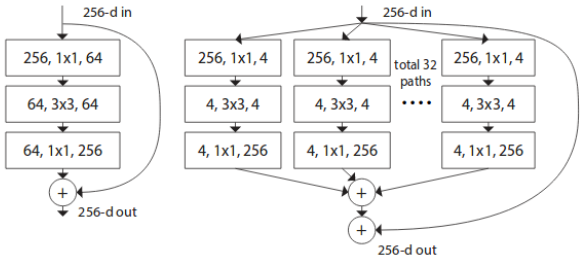

Figure 5. Residual Network

Problem Formulation

In this project, the major task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories. Along with the emotion classification, we intend to advise the individuals experiencing negative emotions with recommendations that could improve their mood. To achieve the latter part of the problem, we develop a Graphical User Interface where the user can input images and correspondingly, recommendations are made based on the emotion of the user.

Data Description

For the classification of the emotions, we used a dataset consisting of facial images pertaining to multiple subjects. Each of the faces is centered and occupies the same amount of space in each image. The dataset is divided into train, validation and test datasets. Each of these dataset have two columns labeled as 'emotion' and 'pixels'. Each row of pixel column is a string with space-separated pixel values. The number of pixels is kept constant in all the images and is equal to 2304 (48 x 48 x 1). Column 'emotion' contains the class or numeric code ranging from 0 to 6 with each code representing an emotion The train dataset has a total of 28709 images, test and validation each having 3589 images.

In the train dataset we can observe the count of each class being represented in Fig.7. We can observe that there are a majority of 'Happy' class samples while 'Disgust' is very less compared to other classes. This disproportion in the dataset makes the machine learning model to learn less

Emotion	Emotion Class
Angry	0
Disgust	1
Fear	2
Happy	3
Sad	4
Surprise	5
Neutral	6

Table 1. Emotion and Numeric Code

Figure 6. Examples of some images in the dataset

about 'disgust' class and more about the classes with more number of data samples.

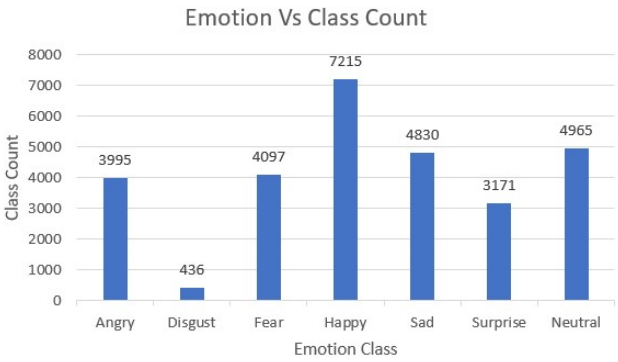

Figure 7. Emotion Vs Class Count

Proposed Solution

In our solution we thought of moving forward with the incremental approach of starting with basic models to using complex pretrained models to test the accuracy of the methods on FER 2013 dataset. As, we had enormous amount of data, we didnt feel the need for data augmentation to increase the size of training dataset. So, we started off with traditional CNN and then incorporated VGG-19, Resnet and Inception V3 into the analysis. After, that we tried the ensemble approach of using different yet complementing models and tried different combinations for best results.

Models Used

- Traditional CNN
- VGG-19
- Mini-Xception
- Inception V3
- Resnet 50
- Ensemble Models

Training Methodology:

As mentioned above that with enormous amount of training and validation data available, we trained each of the mentioned models step by step on the training data and monitored the accuracy of the data on validation set.

We used the pre-trained models for neural networks in this project. Rather than employing the pre-trained weights, we just used the layer architecture of the model and trained it completely on the FER 2013 dataset. So, in our models the weights were trained w.r.t to the FER dataset available to us.

To avoid overfitting, we came up with the approach to monitor train and validation loss in parallel. So, while training the model we trained the each of the model using different number of epochs and validated it on the validation set with the corresponding epoch quantity. A plot of both training loss and validation loss over different epochs indicated the right number of epochs to choose. With a greater number of epochs, the accuracy increases at first, but if we used too many of them, then the train loss surpasses the validation loss. That point is where we decided to stop as beyond that we would be overfitting the data.

We tested each of the model sequentially and monitored the right number of epochs to be chosen.

Network Ensemble

Prior research has suggested that an assembly of multiple neural network models can outer perform an individual model. There are some key points to be taken into consideration before applying the ensemble technique.

- The models should be diverse so that they can complement each other.
- An appropriate ensemble network can effectively aggregate the committee networks.

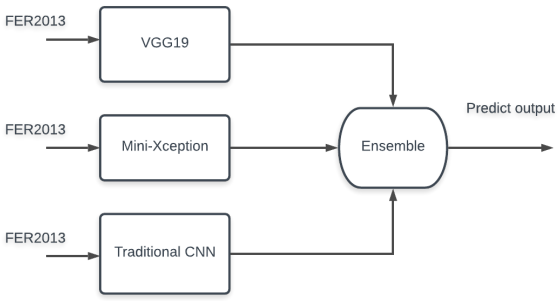

Figure 8. Ensemble

For this we considered models trained on other FER datasets, froze their weights and trained the fully connected layer in the model on the current dataset. This way we made sure that the models were diverse and would complement each other. As, these pretrained models were trained on different types of datasets, but the end goal was the same, to perform facial emotion recognition.

We incorporated Traditional CNN, VGG-19 and Mini-Xception models in the ensemble method and did a decision level ensemble. Where we used stacked generalization to concatenate the output of three models to create one ensemble output.

The figure below is a high-level representation of the ensemble model.

Results

We have trained VGG19, mini-Xception, Simple CNN, Inception-V3, Resnet50 separately. In table 2, we have these different networks and their corresponding accuracy.

The model index 1,2,3... are below models respectly.

1. Ensemble1 (**VGG + Mini-Xception** + Simple CNN)
2. Ensemble2 (**VGG + mini-Xception**)
3. **VGG19**
4. **Mini-Xception**
5. **Traditional CNN**
6. **Inception V3: Top 5 layers**
7. **Resnet50**

We build several ensemble learning networks, the most complex model combines all 5 models we have together.

For the Inception-V3, we take all layers' weights from pretrained using ImageNet, while the last layer's weight we trained using FER2013 data to see what will a transfer learning result be.

We also have different ensemble model that combine VGG, mini-Xception, and/or Simple CNN.

Model	Train	Validation	Test
1	91.7%	66.2%	68.7%
2	92.8%	66%	67.8%
3	97.18%	63.63%	64.78%
4	69.05%	63%	64.08%
5	59.34%	55.11%	55.39%
6	42.73%	40.71%	41.04%
7	45.62%	31.96%	24.38%

Table 2. The train/validation/test accuracy of seven different models

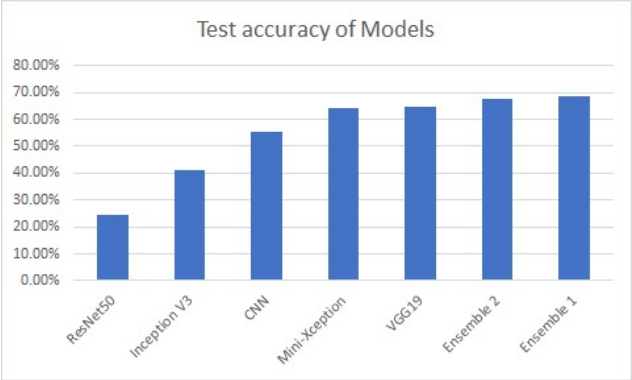

Figure 9. Test accuracy of the models

Future Work

This project of emotion recognition has countless applications and future developments ahead of it. We could think of two most important future developments that could be made possible through the implementation of the model we have built.

Smart Cars

The project could further be developed to take real time image data for the person driving a car and by classifying their emotions, the car can predict if a person is tired or feeling drowsy. This can help prevent fatal accidents due to fatigue or drivers falling asleep. So, here we can imagine that by using this algorithm the car can signal the driver to take a break or have some rest before getting on the road.

Also, this can be used to personalize the experience for the driver depending on their mood. The car can predict the emotion and then change the music and ambient environment accordingly.

Security Systems

This project has deep applications in security monitoring. Where, the algorithm can detect the emotion of a person entering a banking institution and alert the bank representatives about their mental state by predicting their emotions. So, that if a person is feeling anxiety or is angry, they can take countermeasures to handle that.

Conclusion

In this report, we proposed several results comparison for various models. The highest accuracy produced by a stand alone model is 64.78% by VGG19. The complete system used the Typ 1 Ensemble model (VGG19 + MiniXception + CNN) and gives an accuracy of 68.7%.
A simple GUI to this system opens the application possibilities from customer satisfaction analysis to personal computer psychiatrist.

References

[1] Dumoulin V, et. al. "A guide to convolution arithmetic for deep learning", arXiv:1603.07285v2 [stat.ML]

[2] Simonyan K, et. al. "Very Deep Convultional Networks for Large-Scale Image Recognition", arXiv:1409.1556v6 [cs.CV]

[3] Octavio Arriaga, et al. "Real-time Convolutional Neural Networks for Emotion and Gender Classification", CoRR, https://dblp.org/rec/bib/journals/corr/abs-1710-07557, 2017

[4] Szegedy C, et. al. "Rethinking the Inception Architecture for Computer Vision", arXiv:1512.00567v3 [cs.CV]

[5] Szegedy c, et. al. "Going Deeper with Convolutions", https://ai.google/research/pubs/pub43022

[6] Kaiming He, et. al. "Deep Residual Learning for Image Recognition", arXiv:1512.03385v1 [cs.CV]

[7] Andrew M Webb, et. al. "Joint Training of Neural Network Ensembles", arXiv:1902.04422v2 [stat.ML]