Deep Learning (COSC 2779/2972) – Assignment 1 – 2022
Chander Mohan (S3905185)

1. Problem Definition and Introduction:

Introduction:

The objective of this project is to develop a single CNN neural network that will automatically recognize facial expression and facs code for an image. The model will take image as input and gives the output of emotion and facs codes with respect to image. And also we need to develop our own data loader to preprocess the data and make it suitable for model.

Dataset Description:

In this project, we will use CK+ dataset consisting of 560 labelled images from 123 subjects. The dataset contains different types of images like grayscale and RGB images. In our dataset the target variables are high level emotion which has three different categories negative, positive and surprise, and also 15 different facs codes label columns that consist with binary values.

Data Analysis:

When we did some analysis in the dataset then we saw that there are different colour scale images in our dataset. In terms of the distribution of high-level emotion labels, we find that approximately 58% of the images are labelled as conveying a negative emotion, while the remaining images are distributed between the positive and surprise labels. The majority of facs code columns consist of 0. We also plot some graphs to understand the emotion class and facs code distribution.

2. Data Preprocessing:

To prepare our dataset for model training, we did some data preprocessing. Firstly, we split the dataset into three parts: training, validation and test dataset. For our model we need the data in batches and we are using VGG16 so the image size we need is (224,224,3). While creating a data loader we will preprocess the image and resize the dimensions of the image with all the images being equal size. Also there are some images that are grayscale that means they have only one channel so we convert the grayscale images into 3 channels and then we convert it into a numpy array so that is suitable for our model. For the emotion label, they are in categorical form so we convert into numeric values. The output of the data loader after preprocessing the images and labels: (i) numpy array of image (ii) Dictionary in which key value pairs of emotion label and facs code in numpy array.

3. Model Selection, Transfer Learning and Architecture:

Model Section:

For predicting the emotion and facs codes we choose VGG16 to use. We have selected this model because the model architecture is simple and easy to understand as compared to other models like googleNet and ResNet. The VGG16 consists of multiple layers with small 3x3 filters, making it easy to understand and interpret. VGG16 has pre-trained on ImageNet dataset that will provide us a solid starting point for transfer learning and getting the image features. One of the reasons we choose VGG is because it is suitable for small datasets because of its uniform architecture. This helps in preventing overfitting and enables the model to generalize well to the specific task of facial emotion recognition and FACS code prediction, where the dataset is very small.

Transfer Learning:

Our dataset for this task is very small so we choose to use transfer learning. We take the VGG model and freeze its all layers and take the trained weights. This approach increases the model performance and also reduces training time and overfitting risk. After freezing all the layers at the top of the model we attach our own custom layer and two output layers for predicting the emotion class and FACS codes.

Model Architecture:

After retaining the trained feature of VGG16 by freezing all the layers we add our own custom layers. First, we will use flattening the output and after that apply batch normailzation and then again pass through the Dense layer with 512 neurons with ReLU activation function, after that again we are using batch normaization to normalize the activations of the Dense layer. At the end of the model we added a dropout layer that will dead some random neurons and help to reduce the overfitting in the model. And at the end we add two output layers for our prediction.

4. Experiments & Tuning (Justifications):

Training without Regularization Methods:

Initially, I trained the model without any hyperparameter tuning and without any regualization technique on train and validation data. I just add a Dense layer and Adam optimizer while compiling without any custom learning rate then I check the output so I observe that model is overfit and it is memorizing the training data because the accuracy on training data after 30 epochs is 100% and 60% with respect to validation data. So we decided to add some regulaization techniques.

Regularization Methods:

For reduce the overfitting of the model we decide to use some reguralization technique and add these techniques step by step and observe the result and then go back again and experiments with different parameters.

(i) Data Augmentation:

For data augmentation techniques, we add one more feature in our data loader. Data augmentation is a key feature to reduce overfitting. It will perform some transformations on images like rotation, brightness, zoom range, width and height sifting. This technique helps to enrich the training data with variations that simulate real-world scenarios. This approach helps to prevent the model from memorizing the training samples and also helps the model become more robust by learning to generalize across different data instances. I tried with different parameters and selected rotation, brightness, zoom and height-width sifting because if I use a variety of parameters and increase the range of these parameters it will make the image very different with validation data and the model will not perform well on validation data so choose only these transformation techniques.

(ii) Adam Learning Rate:

Choosing a good learning rate is essential for a model. For our model I selected a learning rate 0.001 after the experiments. I used the learning curve with epochs and loss to analyse the effect of learning rate. If I am selecting large learning steps the learning curve fluctuates more and if I choose very small learning steps the model is taking a long time to train and it goes towards overfitting. So after experimenting with different learning rates and observing the learning curve I decide to use the 0.001 learning rate.

(iii) L1_L2 Regualization:

After doing data augmentation and adding learning rate, the model is still overfitted and the learning curve is oscillating. So, I added both L1 and L2 regulaization techniques. Firstly , I experimented with L1. L1 regulaization encourages sparsity by driving some weights to exactly zero and feature selection capability can help eliminate irrelevant features and that also helps for improving the model performance for me. And after that I try with L2 regulaization which takes small weights by penalizing large weight values and it would prevent the model from becoming overly sensitive to individual data points that also helps for my model to improve the accuracy and stable the learning curve so i decide to experiment with both l1_l2 together and I choose 0.001 strength for both L1 and L2.

(iv) Early Stopping:

I also add the callbacks feature while training the model that will help me to prevent from overfitting if i use a large number of epochs. I use 17 patience that means if my validation loss is not decreasing after 17 epochs it will stop training the model.

(v) Batch Normalization layer:

After flattening the output of the VGG16 base, the batch normalization layer is added. Before passing them to the subsequent dense layers, this will normalize the flattened activations.After the first dense layer with 512 neurons and ReLU activation, another batch normalization layer is added. This batch normalization layer normalizes the activations of the dense layer before applying the dropout regularization. It helps stabilize the distribution of activations and can prevent overfitting by acting as a form of regularization.

(vi) Dropout layer:

In this model, we have a large number of parameters so by using a dropout layer it will deactivate some neurons and that will not come into use of forward and backward pass and that will help to prevent overfitting. So I tried with different parameters of the dropout layer and the best performance of the model is on 0.5 with dropout layer.

Tuning with Other parameters:

I also experiment with other parameters like splitting the data ratio for training and validation like I experiment with 60% of training data so I have more data for validation and testing but it will not affect my model's accuracy much. Secondly, I also try with different neurons in my dense layer because sometimes if we use a large number of neurons that will increase the model complexity and that will not be good while training the model and cause of overfitting. Also, I try with different batch sizes but the best number is 32 batch size and I also try with a number of different epochs but when my epochs size is more after 30 it will start to overfit the model. This is because we have a very small dataset.

5. Evaluation Framework

Emotion class Evaluation:

(i) Classification Report: In the context of emotion classification, the Type II error, also known as a false negative because misclassifying emotions can lead to misunderstandings and potentially harm individuals. Looking at the classification report, we observe instances where the model fails to identify true positive cases of certain emotions. For instance, in the positive emotion class, the precision is noted at 0.33, indicating that only 33% of instances predicted as positive by the model are truly positive. The recall, on the other hand, stands at 0.38, signifying that the model captures only 38% of actual positive instances. For the surprise emotion class, the precision of 0.89 indicates that 89% of instances predicted as surprise are indeed surprise instances. However, the recall of 0.57 suggests that the model misses 43% of the actual surprise emotions present in the data.

(ii) Confusion matrix: By analysing the confusion matrix of emotion class, we can say that the model is correctly predicting 47 negative classes out of 57 negative classes. But the model is struggling while predicting positive emotion, mistakenly classifying instances as negative and surprise.

Facs Code Evaluation:

For Facs code we have taken 0.5 as a threshold for predicting a specific Facs Code. For this section, we are doing multi-label classification so accuracy is not a factor to analyse the model performance so we choose F1 macro and F1 micro to analyse.

(i) F1 Micro: F1 Micro aggregating the true positives, false positives, and false negatives across all classes and then calculating precision, recall, and F1-score. Each individual instance contributes equally to the final metrics. For our model the Micro-Precision is 0.50 that means 50% of instances predicted as positive by the model are true positive instances. Micro-Recall is 0.52 that means 52% of the actual positive instances present in the data. Micro-F1 is 0.51 that means that model achieves a moderate balance between precision and recall for all labels combined. That is low because our own data is imbalanced for facs code.

(ii) F1 Macro: In macro-averaging, the calculations are performed at the class level. Each class's metrics are calculated separately, and the macro-average is obtained by averaging these metrics across all classes. For our model macro-Precision is 0.43. This indicates that the model's positive predictions have moderate accuracy on average. macro - recall 0.44 that means the model captures 44% of the actual positive instances present in the data across all classes. Macro f1 is 0.43 is the average F1 score across all the individual classes that suggest that the model achieves a balanced trade-off between precision and recall across all classes.

6. Ultimate Judgment, Analysis & Limitations

We use different techniques to enhance the performance of models and to avoid overfitting. After that the accuracy of the model with respect to emotion class is 71%. The model performs well in terms of precision, recall, and F1-score for the negative class, but has lower performance for the positive and surprise classes. And the relatively low recall of 0.38 indicates that the model struggles to capture a significant portion of the actual positive instances. When performance of a model on multi-label FACS code varies across different labels. Some labels, like AU2, AU17, AU27, and AU25, show relatively accurate predictions, while others exhibit a mix of correct and incorrect predictions. For some labels, the model tends to struggle with false negative predictions, indicating that it misses certain patterns present in the data.

Limitations:

There are some limitations in model and its implementation. The dataset is very limited and there is imbalance in emotion class and also for FACS Codes. So our model is not performing well on FACS codes. To overcome from this problem I explored oversampling within the data loader, I encountered errors during implementation and unable generating new row for the imbalance data by using data augmentation, I also tried with SMOTE technique with a in-bulid library and not able to implement because of this I am not able to balance my data and getting less precision and recall for imbalance facs code labels. when analyzing the accuracy of the validation data across epochs, I observed fluctuations rather than stable trends.. I try to overcome these problems by experimenting and applying different techniques but still there is some problem in the model.

7. Discussion on Ethical issues and biases:

Developing an automatic The development and application of automated facial expression recognition systems raise several ethical issues.

(i) Data privacy: Proper anonymization and privacy protection should be ensured to avoid the misuse of individuals' facial data.

(ii) Cultural Sensitivity: The emotions expressed in the dataset might not accurately represent the emotions across different cultural backgrounds. Applying models trained on this dataset to diverse populations could lead to misinterpretations.

(iii) Fair Representation: The CK+ dataset might not have balanced representation across various demographics, such as age, gender, and ethnicity. Using an imbalanced dataset can lead to biased models that don't generalize well to all groups.

(iv) Labelling Bias: The dataset is imbalanced. 60% of the images are negative labelled images that will lead to Unfair Treatment, Impact on Decision-Making and that will lead to an ethical issue.

(v) Limited Emotion Expressions: The dataset might not include a comprehensive range of emotional expressions, possibly biasing models towards recognizing only a subset of emotions while struggling with more nuanced expressions.

(vi) Freedom of Expression: Misuse of facial recognition in public spaces can deter people from freely expressing themselves for fear of being monitored or judged.

(vii) Manipulation and Exploitation: The technology can be exploited for emotional manipulation, behavioural targeting, or influencing individuals' decisions and emotions without their knowledge.

(viii) Security Risks: If facial recognition databases are compromised, it can lead to identity theft, unauthorized access, and other security breaches.

8. References:

1. MyApps Portal. (2019). Instructure.com. https://rmit.instructure.com/courses/107388/pages/week-6-learning-materials-slash-activities?module_item_id=5060832

2. MyApps Portal. (2019). Instructure.com. https://rmit.instructure.com/courses/107388/pages/week-5-learning-materials-slash-activities?module_item_id=5060830

3. MyApps Portal. (2019). Instructure.com. https://rmit.instructure.com/courses/107388/pages/week-4-learning-materials-slash-activities?module_item_id=5060828