Emotion Recognition in Facial Images using Convolutional Neural Networks

1. Abstract

Facial image recognition refers to the use of computer algorithms to analyze and identify emotions displayed on a person's face.This report explores the problem of facial emotion recognition through deep learning using Convolutional Neural Networks (CNNs). The aim is to classify emotions from facial features such as the position and shape of the mouth, eyes, and eyebrows. Three datasets from Kaggle were selected, ensuring they had enough and different number of classes and images to train the model effectively. The MobileNet v2, ShuffleNet v2, and ResNet 18 architectures were trained on all three datasets and then used to analyze how the combinations of different hyperparameters and models impact the accuracy of facial emotion recognition. ResNet18 resulted as the best model for all three datasets, performing the highest in terms of accuracy and F1-score. FER2013 dataset had the best results overall, likely due to its smaller dataset size and balanced data. MobileNetV2 outperformed ResNet18 and ShuffleNetV2 on FER2013, while ResNet18 achieved the highest accuracy on AffectNetHQ and the Tiny dataset. Lastly in the ablation study, the impact of different hyperparameters on model performance was analyzed. It was found that changing the number of classes for training, the number of images per class, and the learning rate had a significant impact on model performance.

2. Introduction

Facial emotion recognition is the process of using artificial intelligence to analyze and identify emotions displayed on a person's face. Facial expressions are important for conveying emotions and provide insights into a person's feelings that may not be apparent from verbal communication alone. Research suggests that up to 70% of communication is nonverbal, and facial expressions play a significant role in it [1]. By analyzing facial expressions with technology, we can better understand emotional states and improve communication and human interaction. It also has the potential to revolutionize fields like healthcare, education, and marketing. Facial emotion recognition through deep learning aims to classify emotions from an image of a person's face using Convolutional Neural Networks. CNN is used to process the image data and automatically learn and extract relevant features from the facial features, such as the position and shape of the mouth, eyes, and eyebrows. These learned features are then used to classify the person's emotional state. Before the advent of deep learning, facial image recognition faced several challenges, including the lack of robustness in feature extraction and the need for handcrafted feature engineering. Traditional methods relied on extracting and analyzing features such as edges, color, and texture from facial images. However, these methods were often limited in their ability to capture the complex, high-level features needed for accurate facial recognition. These challenges made it difficult to achieve high accuracy rates in facial image recognition. Deep learning methods, particularly Convolutional Neural Networks (CNNs), have significantly improved the accuracy of facial image recognition by automatically learning the most relevant features directly from the images. Unlike traditional methods, which relied on hand-crafted features, CNNs can learn and extract complex features at multiple levels of abstraction from raw images, resulting in better performance. One of the associated challenges with facial emotion recognition is dataset bias, which can occur when the dataset used for training is not diverse enough and not representative of the real-world population, resulting in poor performance and generalization of the model.

Several literature references related to this field were studied, two of the studies that were found particularly relevant were "DeepFace: Closing the Gap to Human-Level Performance in Face Verification" by Taigman et al. (2014) [2], "Facial Expression Recognition with Deep Convolutional Neural Networks" by Liu et al. (2017) [3]. These studies provided important insights on deep learning-based approaches to facial image recognition, specifically on face verification and facial expression recognition using convolutional neural networks. One of the common problems faced in the studies mentioned above was the issue of overfitting, where the model performs well on the training data but fails to generalize to new data. To overcome this, regularization techniques such as dropout and weight decay were used. Another challenge was the class imbalance in the datasets, which can lead to biased models. To address this, some studies used techniques such as data augmentation and weighted loss functions. The choice of appropriate hyperparameters and architecture was also found to be critical in achieving good performance. While

the existing techniques such as using a diverse dataset and using regularization, dropout and cross-validation could improve facial emotion recognition accuracy, they require more computing resources and time for training. One way to address the issue mentioned in facial emotion recognition is by using transfer learning. Use of Transfer learning showed improvement in performance on smaller datasets and reduced the impact of dataset bias. In this study, the problem of imbalanced data was solved by using data augmentation techniques to generate more examples of underrepresented classes. Additionally, the use of transfer learning was explored to leverage pre-trained models and improve the performance of the models.

For the purpose of this project three diverse and annotation-rich datasets for facial expression recognition were chosen: FER2013 [4], AffectNet [5], and FER Tiny [6]. These datasets provided a comprehensive and challenging platform for developing and testing models for facial expression recognition. The pre-processing and filtering techniques employed ensured that the datasets were optimized for efficient training and testing of models. MobileNetV2 [7], ShuffleNetV2 [8], and ResNet-18 [9] were chosen. The models were evaluated based on their computational complexity, in terms of wall clock time required for one-epoch training and the number of floating-point operations (FLOPS) performed during training and validation. The Adam optimizer was used to train all nine CNN models, and the CrossEntropyLoss function was used for multi-class classification tasks. Moreover, the learning rate was tuned to improve the convergence speed and the quality of the final model. Finally, the models were evaluated based on accuracy, precision, F1 score, support, recall, and confusion matrix to assess their performance.

The study found that ResNet18 performed the best on all three datasets, achieving the highest accuracy and F1-score. The FER2013 dataset had the best overall performance, likely due to its well-balanced data and smaller number of classes. MobileNetV2 outperformed ResNet18 and ShuffleNetV2 on all metrics for the FER2013 dataset. While ResNet18 achieved the highest accuracy on the AffectNetHQ dataset, all models performed relatively poorly compared to FER2013. For the Tiny dataset, ResNet18 was also the best-performing model. The study also demonstrated that transfer learning significantly improved the models' accuracy.

3. Methodologies

3.1. Datasets

For this project, three datasets were selected, each with different numbers of classes and varying numbers of images per class. The choice of these datasets is motivated by their large amount of annotations, diversity of the content and classes. All three datasets were obtained from Kaggle.The FER2013 [4] dataset consists of 35,887 grayscale images with a size of 48x48 pixels. Each image is labelled with one of seven emotions, including anger, disgust, fear, happiness, sadness, surprise, or neutral. The dataset was collected from social media websites like Flickr, Google Images, and Yahoo Images, using both human annotators and automated processes.

AffectNet is another dataset of facial expressions that contains over 1 million images with a size of 224x224 pixels. The dataset is labelled with one of 11 emotions, including neutral, happy, sad, surprise, fear, disgust, anger, contempt, valence, arousal, and dominance. The dataset was collected using a combination of human annotators and automated processes to be more diverse and balanced than previous facial expression datasets. A subset of the AffectNet dataset, called AffectNet HQ [6], contains 414,799 high-quality images that were manually selected. This dataset's class representations are more complex than FER2013, with additional classes representing more subtle emotions and affective states, making it a more challenging dataset for building and testing models for facial expression recognition.

The FER Tiny [5] dataset is a publicly available dataset with 42,500 greyscale images of size 48x48 pixels. Each image is labelled with one of thirteen emotions, including anger, disgust, fear, happiness, sadness, surprise, neutral etc. The dataset was collected using a web-based data collection tool, with each image labelled by at least three human annotators. The FER Tiny dataset was designed for training and testing lightweight models for facial emotion recognition on resource-constrained devices, such as smartphones or embedded systems.

Dataset	Total Images	Total Classes	Image Size	Image Format
Facial emotion recognition tiny [5]	15000	13	224x224	Grayscale
AffectNet-HQ [6]	10000	8	224x224	Grayscale
FER-2013 [4]	5000	3	224x224	Grayscale

Table 1. Datasets

To manage the computational complexity and memory usage during training, the sizes of the three datasets were limited, as shown in Tab. 1. Various pre-processing and filtering techniques were applied to standardize the input images and reduce the size of large datasets. Each image in the dataset was resized to a 224x224 resolution and converted to grayscale to standardize the image sizes and reduce the dimensionality of the dataset. To improve model perfor-

mance and stability during training, the mean and standard deviation of the pixel values in each of the datasets were calculated and used to normalize the pixel values in the image dataset. As it can be seen from figure 2 and figure 3, FER 2013 dataset has a slightly higher mean values, indicating that the images in FER 2013 are slightly brighter on average. On the other hand AffectNetHQ dataset has the higher standard deviation indicating that images in the dataset have a wider range of pixel values, and the dataset has more complex features that need to be leanred. Moreover, each dataset was divided into training, testing, and validation subsets, with the training and testing data being split in an 80/20 ratio, and the training data being further divided into an 80/20 ratio for training and validation.

Figure 1. Images from different dataset

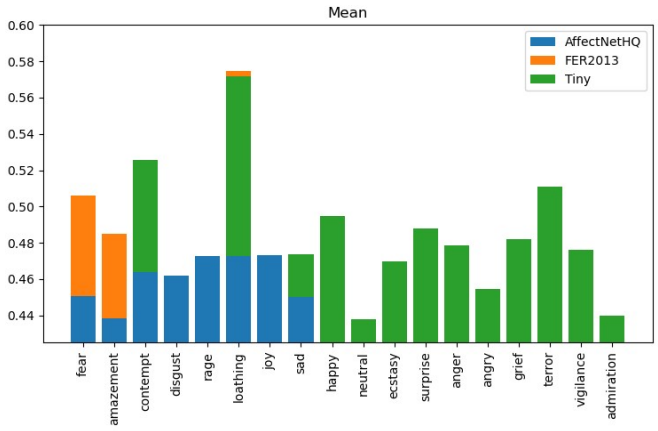

Figure 2. Mean per class

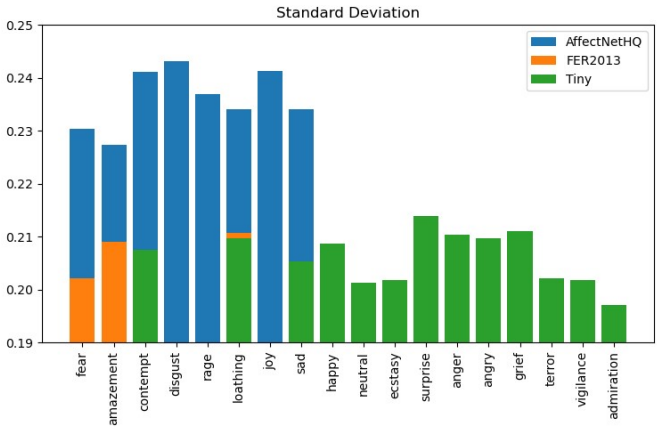

Figure 3. Standard Deviation per class

3.2. CNN Models

MobileNetV2 [7] is a deep learning model that is optimized for use on mobile and embedded devices. Its architecture is based on an inverted residual structure that connects the residual layers to the bottleneck layers, allowing for efficient information flow between layers. The model also employs depth wise separable convolutions, which are used to reduce the computational cost while maintaining accuracy. It contains an initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers. Overall, MobileNetV2 is designed to balance accuracy and efficiency, making it an excellent choice for mobile and embedded vision applications. ShuffleNetV2 [8] is a deep neural network architecture which achieves high accuracy while keeping computational cost low. It uses a channel shuffling operation, enabling information exchange between channels, and depth wise separable and point wise group convolutions to reduce parameters and computational cost. Its architecture includes a stem block, multiple shuffle units, and a fully connected layer. ShuffleNetV2 achieves state-of-the-art performance in image classification benchmarks with small model size.

ResNet-18 [9] is a type of neural network architecture that has been specifically designed to be used for image classification tasks. It comprises a total of 18 layers, which includes a convolution layer followed by four residual blocks. Each block has two or three convolution layers, as well as a skip connection that facilitates gradient flow and a batch normalization layer. To complete the network, here is a global average pooling layer followed by a fully connected layer that handles the classification. ResNet-18 is popular due to its impressive performance in image classification tasks, as well as its relatively low computational cost. The chosen models strike a balance between computational complexity and accuracy, making

them well-suited for various use cases. ResNet18 is ideal for more complex image classification tasks, whereas MobileNetV2 and ShuffleNetV2 are better suited for lightweight and mobile applications. This flexibility enables the selection of the most appropriate model for specific needs while still achieving high levels of accuracy. To further understand the computational complexities of the selected CNN models, they were evaluated based on the wall clock time required for one-epoch training and the number of FLOPS calculation performed during training and validation phases. The metrics for each model is in shown in Tab. 2 .

Architecture Name	Dataset	Time per Epoc
MobileNet v2	FER2013	19.34 seconds
MobileNet v2	AffectNetHq	40.31 seconds
MobileNet v2	Tiny	58.40 seconds
ShuffleNet v2	FER2013	14.53 seconds
ShuffleNet v2	AffectNetHq	30.23 seconds
ShuffleNet v2	Tiny	43.72 seconds
ResNet 18	FER2013	16.00 seconds
ResNet 18	AffectNetHq	34.17 seconds
ResNet 18	Tiny	49.65 seconds

Table 2. Flops and Average Time per Epoc

In terms of computational complexity, ShuffleNet v2 has the lowest number of FLOPS calculations (393.202 billion), making it the most computationally efficient model. MobileNet v2, on the other hand, has the highest number of FLOPS calculations (4018.5 billion), making it the most computationally demanding model. Resnet 18 falls in the middle in terms of FLOPS calculations (990.43 billion). The time required for one-epoch training varied across the models and datasets. ShuffleNet v2 had the fastest training time per epoch for all three datasets as it had the lowest number of flops, whereas Mobilenet v2 took the most amount of time for FER2013 and AffectNetHQ dataset. Overall, the selection of the model depends on the specific use case and the trade-off between accuracy and computational complexity. In terms of computational efficiency ShuffleNet v2 is the best choice. On the other hand, if accuracy is the top priority, then ResNet 18 is the best choice, despite its higher computational demands. MobileNet v2 offers a good balance between the two.

3.3. Optimization Algorithm

In order to train all of our 9 CNN models the Adam optimizer was used [10] which is an adaptive optimization algorithm used in deep learning models, particularly in image classification tasks. It adjusts the learning rate for each parameter based on historical gradient information, allowing for faster convergence and avoiding overshooting or undershooting. It uses first and second moments of gradients for efficient updates and is robust to noisy gradients due to an exponentially decaying average of past gradients and squared gradients. Additionally, it includes a regularization term to prevent overfitting. These features make Adam optimizer a favourable choice for our deep learning image classification models.

To optimize the models during training CrossEntropyLoss [11] function proved to be a good fit for multi-class classification tasks. It computes the negative log likelihood between the predicted class probabilities and the true class probabilities, which penalizes the model for assigning low probabilities to the correct class. Compared to other loss functions like mean squared error or mean absolute error, CrossEntropyLoss function handles imbalanced class distributions well, which is often the case in image classification where certain classes may have more or fewer samples. Secondly, it provides better gradients for backpropagation, which can result in faster convergence and better performance. Overall, these were the reason which motivated the use of CrossEntropyLoss function in this project. While using both of these methods the learning rate and the batch size were fixed to 0.001 and 128 respectively, keeping in mind the available computational resources.

Talking about hyperparameter optimisation, one commonly tuned hyperparameter is the learning rate, which determines the step size at each iteration when updating the model's parameters. The learning rate is a critical hyperparameter as it can greatly affect the convergence speed and the quality of the final model. In this case, the learning rate was tuned by testing a range of values from 0.1 to 0.000000001 on the validation set. After evaluating the model's performance using various learning rates, the best learning rate was found to be 0.0001. Moreover, accuracy, precision, F1 score, support, recall and confusion matrix were used to evaluate the performance of all the 9 models.

4. Results

4.1. Experimental Setup

The facial recognition system was trained and validated with data collected from three different datasets. The training and testing data have been split into a ratio of 80/20 with the training data being further split into an 80/20 ratio for training and validation. To optimize the models, a combination of data augmentation techniques and hyperparameter tuning was used. Transforms to each image in the training set have been added to increase its variability. Additionally, transfer learning has also been used with a pretrained convolutional neural network. A range of hy-

perparameters, including the learning rate, batch size, and number of training epochs were experimented with. Different combinations of hyperparameters were explored and the set of hyperparameters that achieved the highest validation accuracy were selected. To validate the models, their performance was evaluated on the validation set using standard classification metrics such as precision, recall, and F1-score. The final results were achieved with a learning rate of 0.001, batch size of 128, and 25 training epochs as it was found that these hyperparameters provided a good balance between model complexity and generalization performance on our dataset.

4.2. Main Results

In this section, the main results of the experiments are presented, which involved training a total of eleven models using various architectures and datasets. Specifically, nine models were trained on FER, AffectNetHQ, and Tiny datasets using MobileNetV2, ShuffleNetv2, and ResNet18 models, and two models using transfer learning. The performance of each model was evaluated using accuracy and loss metrics on training, validation, and test datasets, as well as confusion matrices for the test datasets.

For the FER2013 dataset, ShuffleNetV2 outperforms ResNet18 and MobileNetV2 on all metrics, likely due to the deeper architecture. ResNet18 achieves an accuracy of 74.12, which is higher than the accuracy of MobileNetV2 and ShuffleNetV2 at 73.43 and 70.69, respectively. MobileNetV2 also has a higher precision, recall, and F1-score compared to ResNet18 and ShuffleNetV2.

On the AffectNetHQ dataset it can be seen from Tab. 3, ResNet18 outperforms MobileNetV2 and ShuffleNetV2 on all metrics likely due to the deeper architecture of ResNet18. However, the performance of all models is relatively poor compared to their performance on FER2013. ResNet18 achieves an accuracy of 58.80, which is the highest among the three models. ShuffleNetV2 and MobileNetV2 achieve accuracies of 46.95 and 48.75, respectively.

AffectNetHQ	ResNet18	ShuffleNetV2	MobileNetV2
Accuracy	0.58	0.46	0.51
Precision	0.59	0.48	0.52
Recall	0.59	0.47	0.52
F1-Score	0.58	0.47	0.51
Train Accuracy	98.22	91.27	81.59
Test Accuracy	58.80	46.95	51.70
Validation Accuracy	55.06	48.81	48.75

Table 3. Comparison between metrics on the AffectNetHQ dataset

ResNet18 is the best performing model on the Tiny dataset likely due to the fact that ResNet18 could capture the features of the dataset more accurately. The model achieves an accuracy of 67.96, which is higher than the accuracies of MobileNetV2 and ShuffleNetV2 at 67.32 and 63.41, respectively.

The graphs in Figures 4 and 5 shows the training and validation loss and accuracy for the resNet18 architecture, showing the performance of pretrained models and the models that were not pretrained. The results indicate that after transfer learning, there was a significant improvement in the accuracy of the model. The pre-trained resNet18-FER model achieved an accuracy of 78.82, compared to 76.18 for the non-pretrained model. Similarly, the pre-trained resNet18-Tiny model outperformed its non-pretrained counterpart, achieving an accuracy of 79.43 compared to 68.33. The graph demonstrates the effectiveness of transfer learning in improving model performance and highlights the importance of leveraging pre-trained models for improved accuracy.

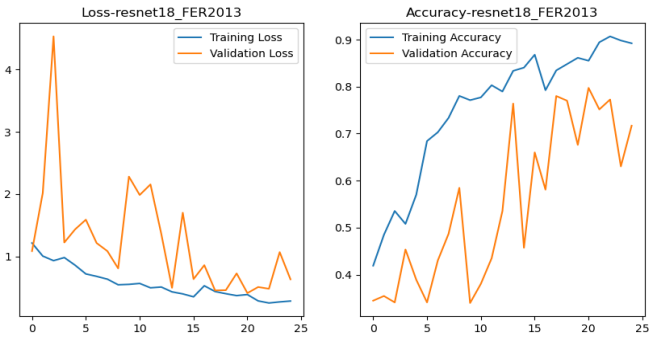

Figure 4. Validation and training loss and accuracy's for FER2013 on ResNet18

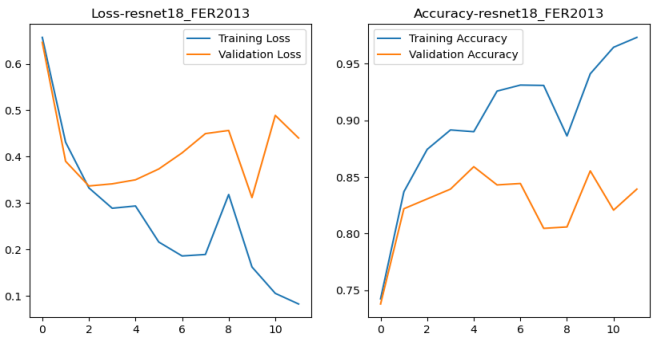

Figure 5. Validation and training loss and accuracy's for FER2013 on ResNet18 pretrained

After implementing ResNet, MobileNetV2, and ShuffleNetV2 CNN models on three different datasets, namely

FER2013, Tiny, and AffectNetHQ, it can be concluded that ResNet18 provided the best performance on the datasets, followed by ShuffleNetV2 and MobileNetV2. The evaluation metrics used to compare the models were precision, recall, F1-score, confusion matrix, and accuracy.The confusion matrix as seen in figure 6 showed that ResNet18 had the highest number of true positive and true negative predictions, indicating better performance in correctly classifying both positive and negative instances compared to the other two models.

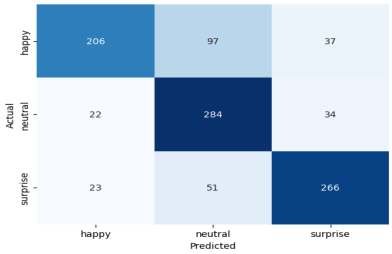

Figure 6. Confusion matrix for the FER dataset trained on ResNet18

As seen in the ROC curve in figure,7, the AUC values were the highest for FER2013 dataset trained on the ResNet18 model, whereas shuffleNetV2 had the least values for AUC, with MobileNetV2 performing slightly better.From the results, it can be observed that the FER2013 dataset had the best results in terms of accuracy, precision, recall, and F1-score, likely because of the larger number of learning examples and well-balanced data. The AffectNetHQ dataset had the lowest performance, with the Tiny dataset performing slightly better. Overall, ResNet18 was the best-performing model, with the highest accuracy and F1-score.

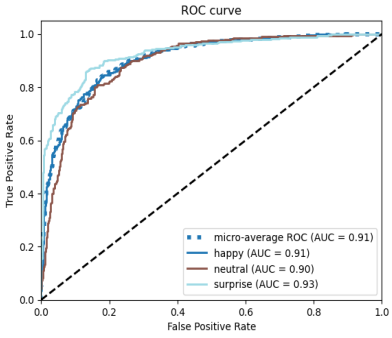

Figure 7. ROC curve for the FER dataset trained on ResNet18

4.3. Ablative Study

After Hyperparameter Optimisation, the highest accuracy was achieved with ResNet-18 architecture on the Tiny dataset with an accuracy of 81.07. For all datasets, a batch size of 128 was used, and after experimenting with a number of epochs, it was found that epochs of 25 works well with our datasets. A learning rate of 0.001 was used after using hyperparameter tuning using grid search to find the optimal learning rate,results of which are seen in figures 8 and 9 and Adam optimizer was used. It was observed that a higher learning rate caused the model to converge quickly to a local minimum of the training loss function, but the model failed to generalize well to new data. To address this, the learning rate was increased from 0.0001 to 0.001. Additionally, it was found that a very large batch size led to overfitting. We experimented with different batch sizes, and a batch size of 128 seemed to work best for all our models. Early stopping was also incorporated to avoid overfitting, requiring the monitoring of validation loss throughout training and terminating the training process when the validation loss stopped improving. Lastly, it was observed that the fine-tuned models attained higher accuracies during the initial epochs, likely due to the advantageous weights they were initialized with, while the deep-tuned models exhibited notably inferior performance compared to the other models, likely due to a modification in the weights of the convolution layers. The ablation study provided further insights into the impact of different hyperparameters on model performance. It was found that changing the number of classes for training, number of images per class, and learning rate had a significant impact on model performance.

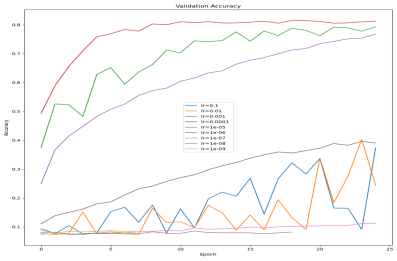

Figure 8. Hyperparameter Optimization validation accuracy for ResNet18 on the Tiny dataset

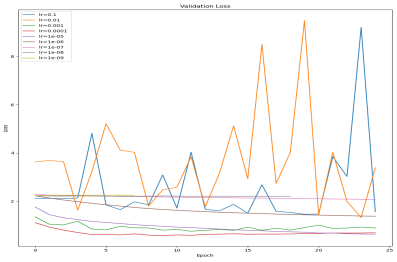

Figure 9. Hyperparameter Optimization validation loss for ResNet18 on the Tiny dataset

References

[1] https://online.utpb.edu/about-us/articles/communication/how-much-of-communication-is nonverbal/. [Online]. Available: https://online.utpb.edu/about-us/articles/communication/how-much-of-communication-is-nonverbal/ 1

[2] [Online]. Available: https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf 1

[3] [Online]. Available: https://www.researchgate.net/publication/321257241_Facial_expression_recognition_using_deep_convolutional_neural_networks 1

[4] https://www.kaggle.com/datasets/msambare/fer2013. [Online]. Available: https://www.kaggle.com/datasets/msambare/fer2013 2

[5] https://www.kaggle.com/datasets/sakuraisana/facial-emotion-recognition tiny?select=validation. [Online]. Available: https://www.kaggle.com/datasets/sakuraisana/facial-emotion-recognition-tiny?select=validation 2

[6] https://www.kaggle.com/datasets/tom99763/affectnethq. [Online]. Available: https://www.kaggle.com/datasets/tom99763/affectnethq 2

[7] https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification 8febb490e61c. [Online]. Available: https://towardsdatascience.com/review-mobilenetv2-light-weight-model-image-classification-8febb490e61c 2, 3

[8] https://sh-tsang.medium.com/reading-shufflenet-v2-practical-guidelines-for-e-fficient-cnn-architecture-design-image 287b05abc08a. [Online]. Available: https://sh-tsang.medium.com/reading-shufflenet-v2-practical-guidelines-for-e-fficient-cnn-architecture-design-image-287b05abc08a 2, 3

[9] https://towardsdatascience.com/understanding-and-visualizing-resnets 442284831be8. [Online]. Available: https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8 2, 3

[10] https://towardsdatascience.com/complete-guide-to-adam-optimization 1e5f29532c3d. [Online]. Available: https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d 4

[11] https://towardsdatascience.com/cross-entropy-loss-function f38c4ec8643e. [Online]. Available: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e 4

5. Supplementary

t-SNE (t-Distributed Stochastic Neighbor Embedding) is a popular dimensionality reduction technique used for visualizing high-dimensional data in a low-dimensional space. As it can be seen from figure 10, figure11, figure12 AffectNetHQ dataset has very less similarities between the features of different classes whereas Tiny and FER2013 dataset have overlapping feature among their class.

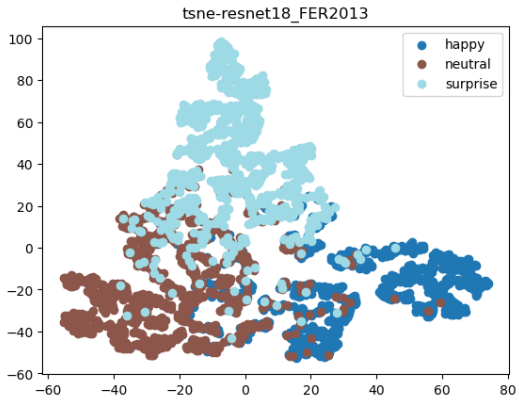

Figure 10. t-SNE Plot for Resnet 18 and Fer2013 Dataset

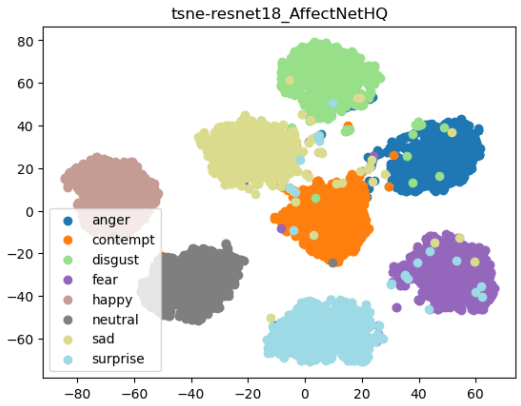

Figure 11. t-SNE Plot for Resnet 18 and AffectNetHQ Dataset

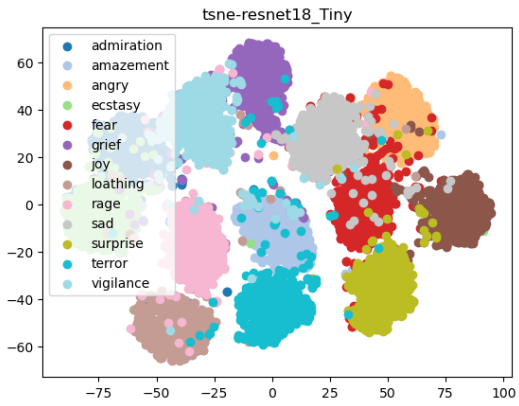

Figure 12. t-SNE Plot for Resnet 18 and Tiny Dataset