Improving Accuracy for Face Emotion Detection

Sayali Lokhande, Bhuvana Ravikumar, Anish Kataria

1 Problem Statement

Face emotion detection is one of the most critical jobs in applications like mental health analysis, human-computer interaction, and surveillance [1]. Despite these advancements in deep learning, many challenges remain, such as improving accuracy given the different facial expressions that may change with light and occlusion, among others. The project uses both ResNet-18 [2] and Tiny-ViT [3] models to show various approaches for improving results in face emotion detection performance. The objective is to evaluate different optimization techniques, find out the best performing technique, and provide a suitable strategy that suits each model.

2 Motivation

Face emotion detection represents one of the very important technologies that bridge human emotion and artificial intelligence together, making machines understand humans more and react to human needs. Emotions form one of the most intrinsic features of human communication, in fact, communicating even more than words. These emotions, if duly detected, can provide a serious set of insights into behaviors and intentions. It cultivates an in-depth analysis of human-to-human interactions that will eventually make any system adaptive and personalized in the process. As devices are increasingly becoming a huge part of our lives, machines can be used interpretedly to make lives even better. Emotion detection would be more accurate and reliable; in this way, AI will be able to cooperate with humans much better and answer both practical and emotional needs. The project's aim is to experiment with various optimization strategies in hopes of finding effective ways of enhancing model performance.

3 Dataset

This project uses the FER2013 dataset [4] for trying out different optimization methods. FER2013 is a very popular benchmark in the face emotion detection area. The images are grayscale and 48x48 pixels large, each picturing a face. The faces have been automatically registered so that the face is more or less centred and occupies about the same amount of space in each image. These images are categorized into seven emotion classes: anger, disgust, fear, happiness, sadness, surprise, and neutral. The training set consists of 28,709 examples and the public test set consists of 3,589 examples.

4 Methods Adopted to Improve Accuracy

- **Learning Rate Adjustment** : In many cases, dynamic adjustment begins with high learning rates then gradually goes down with step-wise refinement in learning progress. Learning rate determines the step size at which the model updates its weights. For this, the project adopted a learning rate scheduler that supports dynamic changes during training.
- **Regularization Method** : Dropout is a powerful regularization method that randomly disables a fraction of neurons during each training iteration. This technique reduces overfitting and improves generalization to unseen data. Weight decay penalizes large weights by adding a term to the loss function, making the model simpler and more reliable.
- **Increased Number of Training Epochs** : Increasing the training time allows the model to learn complicated patterns in the data. However, overfitting should be watched out for, and early stopping can be used where necessary.
- **Data Augmentation** : Data augmentation represents one of the ways of enhancing variety within the training set by randomly rotating, flipping, cropping, and adjusting image colors. It serves as a tool for generalization by introducing the model into various possible real-world environments, minimizing overfitting.

- **Hyperparameter Optimization** : This method focused on refining key training parameters to enhance model performance. Tuning of hyperparameters such as batch size, learning rate, weight decay, and the choice of optimizer is very important to achieve good model performance. This involves trying different combinations in search of the best configuration for the task at hand.

5 Proposed Methodology

- **Data Preprocessing**
  Various preprocessing steps are used in order to make the data compatible and to improve performance during training and evaluation of the model. For both the models, all images are resized to the standard size of 224x224 pixels. After that, the images are first converted into PyTorch tensors, scaling pixel values to the range between 0 and 1. After that, the tensors are normalized with the mean and standard deviation values for the ImageNet dataset: [0.485, 0.456, 0.406] for mean and [0.229, 0.224, 0.225] for standard deviation.

- **Model Architecture**

  1. **ResNet-18**
     ResNet-18 is a convolutional neural network introducing residual connections to solve the vanishing gradient problem for efficient training of deep networks. First, the input image is fed into an initial convolutional layer with a 7 x 7 kernel and stride of 2, followed by a max-pooling layer. The core of ResNet-18 consists of four stages, each containing two residual blocks. Each block includes two convolutional layers, with Batch Normalization and ReLU activation applied after each convolution. Skip connections jump over one or several layers, summing the input of a block with its output to preserve the flow of gradients and improve stability during training. The network increases the number of feature maps progressively from 64 in the first stage to 512 in the last one while using strides of 2 for downsampling. After the residual blocks, global average pooling reduces the feature maps to a single vector; this vector is then input into a fully connected classification layer. A softmax activation function in the output layer generates probabilities for the seven emotion classes.
  2. **Tiny-ViT**
     The model used here is Vision Transformer ViT - B / 16. The input image is split into fixed-size patches, 16×16. Each patch is flattened into a vector and linearly transformed into a fixed-dimensional embedding. These patch embeddings are then combined together with learnable positional embeddings to retain spatial information. A series of transformer encoder blocks form the core of this model. Each block consists of Multi-Head Self-Attention, which captures long-range dependencies between patches; Feed-Forward Neural Network, which applies non-linear transformations to enrich the features; Layer Normalization, which improves training stability; Residual Connections that preserve information and ensure efficient gradient flow; and a Classification Head. A special token, [CLS], is added to the input sequence, which represents the whole image. After passing through the transformer blocks, the final output of the [CLS] token feeds into a fully connected layer, also known as a classification head, to perform label prediction.

- **Approaches to Improve Accuracy**

  1. **ResNet-18**
     (a) **Learning Rate Adjustment** : The learning rate was initialized at 1e-3. Adam optimizer was utilized for its adaptive learning capabilities. A StepLR scheduler was incorporated to adjust the learning rate periodically during training. The step size was set to 7, with the learning rate being multiplied by 0.1 at each step (gamma = 0.1).
     (b) **Regularization Method** : For this model, two main regularization techniques were implemented: dropout and L2 regularization (weight decay). A dropout layer was added to the network to introduce stochastic noise during training. Additionally, the Adam optimizer was configured with a weight decay parameter of 1e-4 to apply L2 regularization, penalizing large weights and encouraging better generalization.
     (c) **Increased Number of Epochs** : Increasing the number of epochs from 30 to 50 allowed the model to undergo extended training.

(d) **Data Augmentation** : Various augmentation techniques were employed to diversify the training data and reduce overfitting. These included random horizontal flipping with a probability of 0.5, random rotations of up to 15 degrees, and adjustments to brightness, contrast, saturation, and hue using `ColorJitter`. All images were resized to 224x224 and normalized with the standard mean values [0.485, 0.456, 0.406] and standard deviations [0.229, 0.224, 0.225].

(e) **Hyperparameter Optimization** : The Adam optimizer was replaced with SGD, leveraging a momentum value of 0.9 to stabilize training dynamics and improve convergence. The batch size was increased from 32 to 64, which enhanced gradient stability and reduced fluctuations during the training process. Additionally, the learning rate was adjusted to 0.01 for SGD, striking a balance between faster convergence and training stability.

2. **Tiny-ViT**

(a) **Learning Rate Adjustment** : The learning rate was initialized at 1e-5. A small value was chosen to ensure stable convergence. Adam optimizer was used because it adapts the learning rate of individual parameters. A StepLR scheduler was incorporated to adjust the learning rate periodically during training. The step size was set to 10, with the learning rate getting multiplied by 0.1 at each step (`gamma` = 0.1).

(b) **Regularization Method** : In this model, two key types of regularization were performed, including dropout and L2 regularisation with weight decay. In the classification head, a dropout layer was introduced in the Tiny ViT model with a dropout rate set to 0.3. With the Adam optimizer, a weight decay parameter equal to 1e-4 was set in order to introduce L2 regularization.

(c) **Increased Number of Epochs** : Increasing the number of epochs from 30 to 50 allows the model to make more passes through the dataset.

(d) **Data Augmentation** : Several augmentation techniques were used, including random horizontal flipping, random rotation of up to 20 degrees, and random resized cropping. Besides, color jittering was applied to adjust brightness, contrast, saturation, and colour. All images were resized to 224x224 and normalized with the standard mean values [0.485, 0.456, 0.406] and standard deviations [0.229, 0.224, 0.225].

(e) **Hyperparameter Optimization** : The learning rate was decreased to 1e-5 to allow for finer weight updates during fine-tuning. A reduced batch size of 16 was selected. Besides, the optimizer was changed to SGD with a momentum value of 0.9.

6 Training

We used the pretrained ResNet-18 [2] and pretrained Tiny-ViT (ViT-B/16) model [3] by freezing all the layers of the pretrained model to extract the features and modified only the classifier according to the seven classes. We then used the weights of this pre-trained model to finetune our models for all the five approaches.

7 Results

Table 1 shows the loss as well as training and testing accuracies for all approaches of both the models.

8 Comparison

ResNet-18 and Tiny-ViT each excel under different conditions. ResNet-18 achieves its best results with data augmentation and hyperparameter tuning, reaching 68.60% testing and 99.71% training accuracy, demonstrating strong generalization. However, it shows limitations with extended training or learning rate adjustments.

Tiny-ViT, on the other hand, performs well with regularization, achieving 67.55% testing and 99.05% training accuracy. However, it struggles with data augmentation and longer training, where testing accuracy drops below 56%. Both models have unique strengths, with ResNet-18 excelling in generalization and Tiny-ViT being lightweight and resource-efficient, making the choice task-dependent.

Model	Method	Loss	Training Accuracy	Testing Accuracy
ResNet-18	Learning Rate Adjustment	0.046	99.83%	66.31%
	Regularization	0.2816	90.11%	63.21%
	Data Augmentation	0.4318	84.28%	68.60%
	Hyperparameter Optimization	0.0059	99.71%	67.48%
Tiny-ViT	Learning Rate Adjustment	0.0036	99.82%	67.25%
	Regularization	0.02	99.05%	67.55%
	Data Augmentation	1.15	57.12%	55.23%
	Hyperparameter Optimization	0.557	81.2%	63.82%

Table 1: Results of Different Optimization Methods on ResNet-18 and Tiny-ViT

9 Conclusion

The comparison between ResNet-18 and Tiny-ViT highlights how advanced optimization techniques can significantly improve the accuracy of face emotion detection models. ResNet-18 is very good at generalization, especially when enhanced by techniques such as data augmentation and hyperparameter optimization, which brought very impressive results for both training and testing. On the other hand, Tiny-ViT is very successful in the case of regularization techniques, turning out to be a lightweight but capable alternative for certain use cases. That said, both models face certain challenges, such as sensitivity to longer training periods or particular augmentation strategies. These findings indicate the importance of tailoring optimization methods to suit a particular model architecture. This, in turn, points out that achieving reliable and robust performance in emotion detection is highly dependent on finding the right balance and adapting strategies to the task at hand.

10 Contributions

- Sayali Lokhande - Implemented all the five approaches for improving accuracy of TinyViT model.
- Bhuvana Ravikumar - Implemented all the five approaches for improving accuracy for ResNet-18 model.
- Anish Kataria - Implemented the base model for ResNet-18 and Tiny-ViT and worked on data preprocessing.

References

[1] https://visagetechnologies.com/facial-emotion-recognition-guide/

[2] https://pytorch.org/vision/main/models/generated/torchvision.models.resnet18.html

[3] https://huggingface.co/google/vit-base-patch16-224

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