CNNs for Facial Emotion Classification in Color and Grayscale

Anusree Chittineni, Benjamin Fargnoli, Anjali Kolapalli,
Kasey Lowe, and Mackenzie Williams
College of Computing
Georgia Institute of Technology

Abstract

Emotion recognition is an application of image classification, often used in industries like mental health monitoring and evaluation of customer satisfaction. This study bases its foundation on the standard use of grayscale images in most modern AI/ML softwares. Due to the role that skin colorization plays in portraying emotion, such as blushing or flushing of the skin, there may be added value to using colorized images alongside the added computational costs. Additionally, different models may respond differently to these visual cues, so this experiment tests on ResNet, DenseNet, VGG16, and EfficientNet. Experimental results support these hypotheses as each model shows improved accuracy when using colored images, however the accuracy percentage difference between black and white and colored trials across models ranges from 0.9% to 10.5%.

1 Introduction

Emotion recognition from facial expressions has emerged and is now an important topic in computer vision, with applications that range from human-computer interaction and health diagnostics to driver sentiment and fatigue detection. The task at hand involves recognizing an individual's emotional state, such as anger, sadness, or happiness, based on visual input from facial images.

Although applications of emotion detection are growing and developing, there is still a gap between research and real-world use, especially in the type of images that are used. Many training data sets, such as FER-2013, are grayscale, but most real-life images from phones, webcams, CCTV, etc. are usually in color. This disparity leads us to an important question: do certain emotion detection models perform better with grayscale images or with colored ones, and if so, under what circumstances is this difference noticeable and/or significant?

To understand and answer this question, our project compares how well several machine learning models can perform when they are trained and evaluated using grayscale facial images and colored facial images. We used a subset of the AffectNet dataset, which includes eight key facial expressions: Anger, Disgust, Fear, Sadness, Happiness, Surprise, Neutral, and Contempt. Each model was put through two training and evaluation phases: one using the original colored images and one with the grayscale versions. The performance of each model across both datasets was then compared to see how much difference color really makes in the results.

Our team evaluated four convolutional neural network (CNN) architectures - ResNet18, DenseNet121, VGG16, and EfficientNet B0 - to find out how each model reacts to color in facial emotion images. We chose these different models because they all work in different ways which gives us a range of results. We not only measured accuracy but we also measured how the models learned over time and where certain errors were made.

CS 4641: Machine Learning, Georgia Institute of Technology. (LaTeX template from NeurIPS 2020.)

Our findings offer data-driven insights into how color in facial emotion recognition affects its classification and also helps identify which models can provide the best results for real-world emotion detection systems.

2 Background

Our work draws on and builds upon a number of similar studies from recent years. A 2018 study from Stanford called "Image Colorization and Classification" went into depth about adding color to black and white images and how it could help certain models perform (Li & Wang 2018). Their results show that some models, like the VGG-style networks, performed better with color. Others, such as the regression models, did much better with the grayscale images [1]. These findings show that colorization can only improve certain models. However, it is important to note that these findings are for images that were originally black and white and were artificially colorized, as opposed to images that were originally in color.

In 2024, Abdulameer et al. (2024) conducted a study in the area of medical imaging and compared how CNNs performed on grayscale versus color images for malaria cell classification. They found that the models that trained on color images resulted in higher accuracy (98.25%) than those that trained with grayscale images (95.72%) [2]. Even though this research was based on medical data rather than emotions, their findings still show that color can help models discover useful details and improve their performance.

In contrast, another study called "What's color got to do with it? Face recognition in grayscale" provides much different results. This study found that for the most part, using black and white images instead of color did not significantly impact the performance of deep learning models in facial recognition [3]. However, the researchers did note that deeper models performed just as well with or without color but smaller models had slightly increased accuracy from using colored images. The results from this study suggest that a model's complexity can play a major role in whether color impacts its performance or not.

Previous work in facial emotion recognition such as Khajuria et al. (2023) has shown that CNNs and VGG16 models achieve higher performance on emotion recognition or classification tasks compared to other models [4]. However, these studies do not examine the impact of color or black and white images on these models, which is what our team hopes to address.

Although the AffectNet dataset is commonly used for emotion classification, research is typically focused on using color or grayscale images, but not both. Our approach is novel because we are comparing the color and black and white versions of the same dataset across a number of model architectures, which will give us a direct analysis on how color affects the performance of each model differently. Additionally, previous studies produce conflicting results as to the impact of color on CNN models, a question to which we hope to gain more insight on answering.

3 Methods

3.1 Overview of Proposed Approach

To learn more about how color really affects emotion detection, we set up an experiment using deep learning models which are trained on both color and black and white images. As mentioned before, our team used a smaller subset of the AffectNet data set to make two sets of images: one with its original color format and the other converted to grayscale using OpenCV. We made sure that color was the only difference in the test by only changing the type of image.

We trained the models, each with different design and complexity components, on both of the image types. Each of the four models were trained using the same type of data, settings, preprocessing techniques, and evaluation metrics. This type of setup lets our team clearly compare how much color makes a difference in the performance of the models without confounding variables.

3.2 Motivation for Architecture Selection

To understand how the design of the model impacts color sensitivity, we chose four CNN models that vary in depth, complexity, and design aspects:

- **ResNet18**: This is a smaller model with 18 layers. It uses skip connections to avoid certain issues, like vanishing gradients. It is better for capturing mid-level image features so we can see how color changes its performance.
- **DenseNet121**: This is a deeper network where all of the layers are connected to one another. This helps the model reuse features and focus on the minute details– useful for detecting emotions in both color and black and white images.
- **VGG16**: This is a well-known model made from basic 3x3 convolutional blocks. It is normally used in image recognition tasks and this gives us a valuable benchmark for a task.
- **EfficientNet B0**: This is a highly optimized model with 237 layers that balances network depth, width, and resolution by using compound scaling. It also provides advanced components like MBConv and squeeze-and-excitation blocks, which makes it more efficient and accurate for emotion recognition. It's speed and efficiency make it good for real-world emotion detection.

3.3 The Data

The data used to train and test the models came from the affectnet-yolo-format dataset on Kaggle, which is a subset of the AffectNet dataset introduced in Mollahosseini et al. 2019 [6]. It consists of just over 25k 96x96 color images of facial expressions, as well as corresponding labels denoting the emotion displayed (Anger, Contempt, Disgust, Fear, Happy, Neutral, Sad or Surprise). We used openCV to make a copy of this dataset with every image converted to grayscale.

```
# Class labels map
labels_map = {
    '0': 'Anger',
    '1': 'Contempt',
    '2': 'Disgust',
    '3': 'Fear',
    '4': 'Happy',
    '5': 'Neutral',
    '6': 'Sad',
    '7': 'Surprise'
}
```

Figure 1: Emotion Mapping

```
Class: Fear - Images: 2021
Class: Contempt - Images: 1996
Class: Happy - Images: 2154
Class: Anger - Images: 2339
Class: Neutral - Images: 1616
Class: Surprise - Images: 2819
Class: Disgust - Images: 2242
Class: Sad - Images: 1914
```

Figure 2: Emotion Distribution

Figure 3: Example of Image Inputs

3.4 Training Procedure

Our team trained all the models used with the same data split: 70% for training, 20% for validation and 10% for testing. This resulted in around 17.5k images and labels in the train set, 5k in the validation set, and 2.5k in the test set. After initial testing on all models using the same hyperparameters, the learning rate and number of epochs were adjusted for each model in order to optimize their performance. We added L2 regularization, also known as weight decay, and also used a learning rate scheduler to reduce the learning rate if validation performance stopped improving.

Our team used categorical cross-entropy as our loss function initially because it is suitable for multi-class classification. Models updated their weights using backpropagation based on the loss between the predicted outputs and the true labels throughout training. To determine the most effective model, we utilized early stopping and check pointing mechanisms based on validation metrics.

3.5 Evaluation Metrics

We used a few key evaluation tools to measure how well each model worked with both color and black and white images.

- **Accuracy**: The percentage of correct predictions on the test set was calculated using `sklearn.metrics.accuracy_score`.
- **Mean Test Loss**: The average error (cross-entropy loss) the model made on the test set, which helps us really understand how confident and accurate the predictions were.
- **Confusion Matrix**: This is a detailed chart showing which emotions were confused with others, helping us to see whether color reduced those errors.

Each model was trained independently on both input types. We not only compared the overall performance of each model, but our goal was to analyze whether certain architectures—based on connectivity or depth—gained more accuracy from color inputs.

All code used for preprocessing, training, and testing can be found at this Github repository.

4 Experiments

Experimentation first consisted of preliminary testing of black and white versus colored data sets on the four models using standardized hyperparameters, including an Adam optimizer, learning rate of 1e-5, batch size of 64, weight decay rate of 1e-4, and 20 epochs. Initial findings indicated that the colored data set led to an average test accuracy increase of 4.32% across all models, however overall accuracy in all trials ranged from 60.51% to 71.77%, rates we hoped to improve before obtaining final values. These early results provided a sense of direction for continued model-specific tuning as well as a foundation for the type of results that were expected to become more prevalent upon improvement of the models. Upon tuning the models independently of each other, we used the hyperparameters for each model that resulted in the best overall testing accuracy, ensuring that each model's testing was a result of its best performance.

Table 1: Experimentation Final Results

CNN Model	% Accuracy on BW Data	% Accuracy on Color Data
ResNet	68.78%	71.29%
DenseNet	67.65%	71.7%
VGG	70.83%	71.77%
EfficientNet	71.68%	71.86%

4.1 ResNet

With the initial standardized hyperparameters, ResNet18 performed with an accuracy of 60.51% on black and white data and 71.37% on colored data. Upon tuning the parameters based on the information collected from referencing various published papers, the black and white results were improved by about 8%. The colored result declined by .08% which is insignificant enough that the newly tuned hyperparameters are what we are considering the best performance for Resnet18. The model has been tuned to use the parameter set up of the paper "Deep Residual Learning for Image recognition" [5]. This paper utilizes SGD as the optimizer along with a learning rate scheduler. The learning rate starts from 0.01 and is divided by 10 every 5 epochs. We ran the model for 20 epochs to get the results shown below. We kept a weight decay of 0.0001 and a momentum of 0.9 for the optimizer. Additionally, we removed the dropout layer because it was not utilized in the base paper.

Prior to referencing the paper of He et al. [5], further experimentation with Adam as the optimizer was conducted. We attempted to tune by adjusting the learning rate without implemented a lr decay. The results of this tuning all point towards signs of overfitting. Past 10 epochs, the model's validation loss would consistently start to curve back up by about 0.2 until eventually plateauing around a loss of 0.8 while the training loss would continue to decline. A shift over to resnet50 to test the theory of shallow versus deeper networks also showed no sigs of overcoming the overfitting issue.

Implementing SGD, as opposed to Adam, showed signs of improvement among the black and white data, but still experienced a trend towards overfitting.

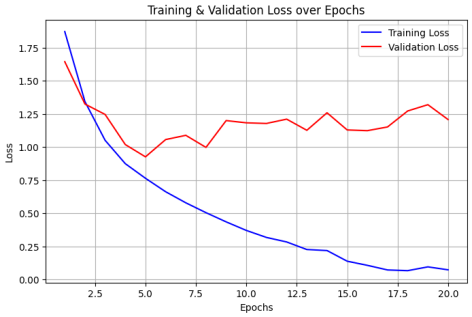

Figure 4: Training and Validation Loss for ResNet18 on Black & White Data

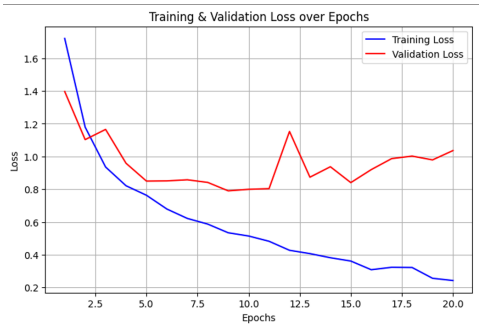

Figure 5: Training and Validation Loss for ResNet18 on Colored Data

4.2 DenseNet

After tuning the DenseNet model, we found that the optimal hyperparameters seemed to be the ones used in the preliminary testing: a batch size of 64, learning rate of 1e-5, and weight decay rate of 1e-4. However, we found that using Adam as the optimizer instead of SGD increased the model's performance. Under these conditions, the model achieved a test accuracy of 71.7% on the colored dataset and 67.65% on the black and white dataset, meaning it performed better on the color data. This difference was seen across all sets of hyperparameters we tested, as in every trial, the model performed 3-5% better on the color data.

It should be noted that the DenseNet model, like some of the others, consistently struggled with overfitting. When trained for 20 epochs, the model's train loss consistently decreased throughout, but its validation loss leveled out and even started to increase in some cases around the 10 epoch mark, suggesting that the weights it began learning were specific to the training dataset. We attempted to address this by adding dropout but found that it had little effect, which leads us to believe that the overfitting is largely due to the relatively small size of the dataset.

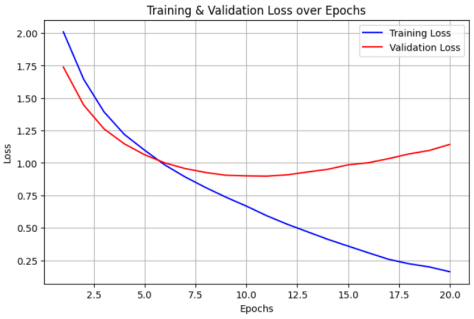

Figure 6: Training and Validation Loss for DenseNet on Black & White Data

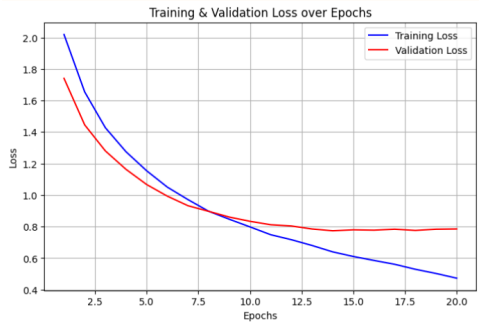

Figure 7: Training and Validation Loss for DenseNet on Colored Data

4.3 VGG

Like with DenseNet, we concluded that the best hyperparameters for the VGG model were the originals. Although we tested changes to learning rate, weight decay, and number of epochs, we were unable to create statistically significant change in our VGG performance. This led to final results of 70.83% test accuracy on the black and white data set and 71.77% on the colored data set. Although we were unable to see model improvement, these were consistently the most accurate results in

preliminary testing, suggesting that the original hyperparameters may have already been most suitable for this type of model. Like DenseNet, the VGG model also showed signs of overfitting.

Figure 8: Training and Validation Loss for VGG on Black & White Data

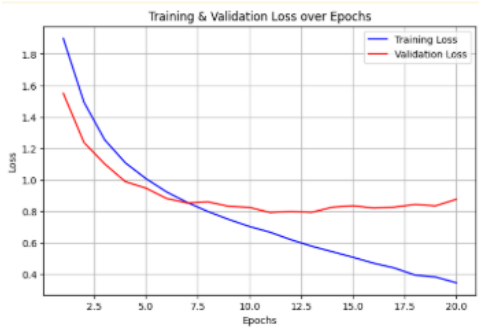

Figure 9: Training and Validation Loss for VGG on Colored Data

4.4 EfficientNet

In order to tune the EfficientNet model, the first change made was swapping the SGD optimizer for the AdamW optimizer. Because the AdamW optimizer works better with sparse data sets, this led to an average of a 4% accuracy increase across EfficientNet trials, even without implementation of other changes. In tuning the learning rate, it was discovered that the learning rate of 0.0003 led to the most accurate model. Any lower seemed to increase training loss and any higher seemed to decrease training loss, but lead to higher val loss, causing slight overfitting of the model. We also tried data augmentation as well as another loss function, though neither contributed to model performance. In the end, this model ran the AdamW optimizer with a learning rate of 0.0003 and a weight of 1e-4 across 5 epochs for both the colored dataset and black and white dataset, totaling 10 epochs. Typically, the best model emerged around the second or third epoch and, similarly, this is where validation loss reached its lowest point and often showed stabilization (Figure 11, Figure 10).The final results from the EfficientNet trials were 71.68% testing accuracy on the colored data and 71.86% testing accuracy on the black and white data set. While this accuracy is lowered than desired, it may be due to the architecture of EfficientNet, as it's made for real-time analysis and quicker runs as compared to some of our other models.

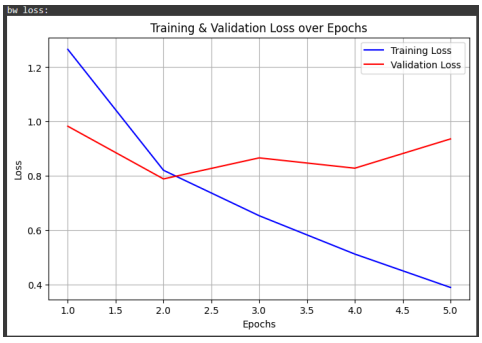

Figure 10: Training and Validation Loss for EfficientNet on Black & White Data

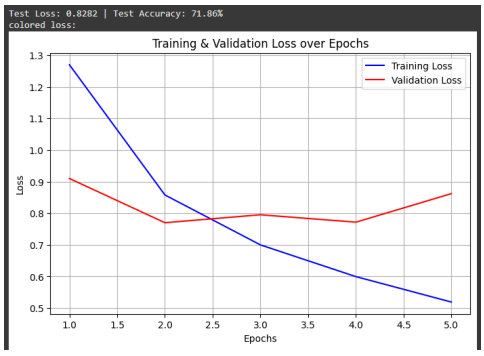

Figure 11: Training and Validation Loss for EfficientNet on Colored Data

Once we achieved optimal performance for each model, we generated confusion matrices using sklearn. However, all four models had comparable confusion matrices, and there were no clear patterns in the errors the models made. Examples of these confusing matrices are shown in Figure 12 and Figure 13.

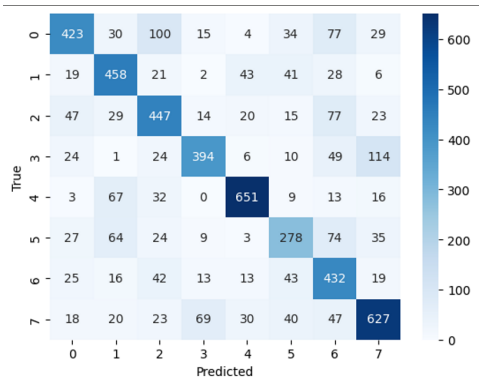

Figure 12: ResNet18 BW Confusion Matrix

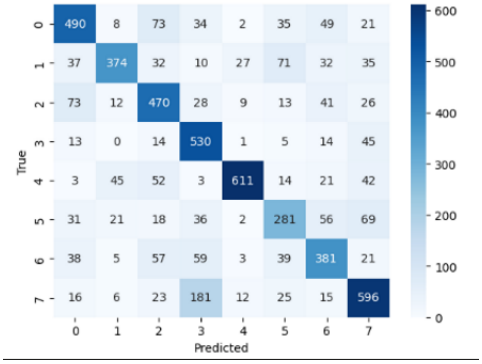

Figure 13: ResNet18 Color Confusion Matrix

5 Conclusion

One main takeaway from our results is that every model performed at least slightly better on the color dataset than on the black and white one, which suggests that in general, the inclusion of color is beneficial to most models' performance. These differences in performance were greatest for DenseNet and ResNet, and were not as pronounced for VGG and EfficientNet. However, all models achieved comparable performances on the color data, so these differences stem mainly from the fact that ResNet and DenseNet performed slightly worse on the black and white data than VGG and EfficientNet. Overall, EfficientNet performed the best on both datasets.

It is important to note that all of the models seemed to be overfitting, despite our attempts to avoid it. We believe this is due to the relatively small size of the training set. This overfitting means the models likely did not reach their optimal performance, so any conclusions drawn based on these results should account for the likely difference in potential performance and experimental results. It can also be noted that the models all had the highest misclassification rates between fear and surprise. Future research on this topic should use a larger dataset so that the models can be tuned correctly in order to accurately compare their performance across the different datasets.

6 References

[1] Li, P. & Wang, Z. (2018) Image Colorization and Classification. `https://cs230.stanford.edu/files_winter_2018/projects/6940128.pdf`. Accessed: Apr. 14, 2025.

[2] Abdulameer, M., Behadili, S.F., Khalaf, A.A., & Radhi, A.M. (2024) Model performance evaluation for color and grayscale images in malaria classification using deep CNN. *AIP Conference Proceedings*, **3219**:030008–030008. 10.1063/5.0237322.

[3] What's color got to do with it? Face recognition in grayscale. (2016) `https://arxiv.org/html/2309.05180v2#bib`. Accessed: Apr. 14, 2025.

[4] Khajuria, O., Kumar, R., & Gupta, M. (2023) Facial Emotion Recognition using CNN and VGG-16. In *Proc. 2023 Int. Conf. on Inventive Computation Technologies (ICICT)*, pp. 472–477. 10.1109/icict57646.2023.10133972.

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016) Deep Residual Learning for Image Recognition. In *Proc. 2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, pp. 770–778. 10.1109/CVPR.2016.90.

[6] Mollahosseini, A., Hasani, B., & Mahoor, M. H. (2019) AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild. in *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18-31. 10.1109/TAFFC.2017.2740923.