A Comparative Study of HOG, CNN, and Residual Network for Facial Expression Analysis

Esha Wang, M.Sc. Stanford University

Abstract – **Convolutional neural networks are the premier image-recognition algorithm, yet they demand intensive computation and large amounts of data. In a low-resolution environment, how these networks fared against alternative algorithms and feature-selection methods was explored. The performance of four machine-learning algorithms in classifying facial emotions on Kaggle's FER2013 dataset were compared, specifically logistic regression, SVC, CNN, and ResNet. In addition to the raw images, HOG variants were used in testing the non-neural network algorithms. With raw pixel inputs, the logistic regression algorithm achieved 34.8% accuracy, while the SVC with an RBF kernel achieved 41.5%. With HOG images, they scored 40.2% and 41.5% accuracy, respectively. A baseline CNN model achieved 40.5% accuracy, while the best-performing CNN model (ResNet50 with image augmentation) achieved 66.1%. The latter model ranked #5 top accuracy on the 2013 Kaggle leaderboard.**

I. Introduction

Over the past decade, facial recognition and analysis have come to the forefront of applied machine learning. These algorithms allow computers to interpret media in ways previously relegated to manual labor, dramatically increasing the speed of photo and video processing. Popular applications include facial recognition, the classification of expressions, and lie detection. For this study, the input to the algorithm is a 48×48 grayscale image, and the output is a facial expression denoted by a number 0-6: (0) Angry, (1) Disgust, (2) Fear, (3) Happy, (4) Sad, (5) Surprise, or (6) Neutral. The objective is to maximize accuracy of the predictions on the test set.

The logistic regression model and support vector classifier (SVC) used in this study were built to work for both this project and Andrew's concurrent CS221 project. All other models were built solely for this study.

II. Related Work

Facial expression recognition is a widely studied machine learning problem, with much research effort currently being dedicated to the area. In 2012, Krizhevsky, et al. designed AlexNet, which became a breakthrough in computer vision [1]. AlexNet won first place at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), obtaining an error of 15.3%, compared to the 26.2% error obtained by the second place entry. AlexNet is a convolutional neural network (CNN) consisting of eight layers total, with five convolution layers and three fully connected (FC) layers. Max pooling was used as the activation function for some of the convolution layers. The model was trained using multiple GPU support with CUDA [2]. The use of CUDA with single GPU support was adapted into the deeper models outlined in this report.

After the success of AlexNet, a huge wave of CNN models were submitted in the 2013 ImageNet Challenge. That year, the winners Zeiler, et al. presented ZF Net, whose architecture was built off of AlexNet with additional fine-tuning and performance improvements [3]. Unlike AlexNet, which trained on 15 million images, ZF Net was trained on only 1.3 million. Furthermore, while AlexNet used an 11×11 filter size on its first convolution layer, ZF Net adopted a 7×7, allowing the algorithm to retain more of the pixel information in the original image. Additionally, ReLU was used as the network's activation function, and the model minimized cross-entropy loss using stochastic gradient descent (SGD).

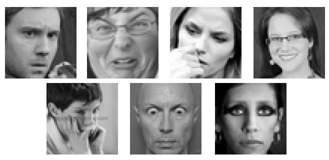
Attempting to build upon the success of AlexNet and ZF Net, Simonyan, et al. investigated CNN depth and its effect on training and test accuracy for the 2014 ImageNet Challenge. Their model consisted of a 224×224 RBG image input, nineteen convolution layers with 3×3 filter size (stride and padding of 1), 2×2 max-pooling (strides of 2), and finally, three FC layers and a softmax layer. Each hidden layer was equipped with a ReLU activation function as well. Simonyan's method, known as VGG, outperformed AlexNet by a substantial margin with a nearly 10% decrease in test error.

Following these breakthroughs, deep CNNs became arguably the most groundbreaking work in computer vision and image classification in recent years. In 2016, He, et al. presented a residual learning framework targeted at deep, difficult-to-train neural network models [5]. While they showed that excessive layer stacking in traditional CNNs led to accuracy saturation, and ultimately degradation, the formulation of feedforward residual mappings allowed for increased depth and ease of optimization. The ResNet model used up to 152 layers and achieved 3.57% error on the ImageNet test set with less computational complexity than VGG neural nets. ResNet has quickly become a monument among CNN architectures, both for its innovation and demonstrated results.

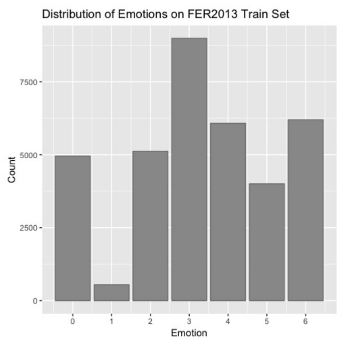
Regarding non-neural network algorithms, few recent papers have conducted the type of comparison performed in this report. One study by Zhou, et al. runs an SVC (using features identified by a neural network) against a CNN in identifying photos of locations labeled by type (windmill, soccer field, elevator, etc.) [6]. Though the authors spend little space discussing the SVC, they report it achieving just over 40% accuracy, almost twenty points behind various high-performing neural networks. This provides a rough baseline estimate for the discrepancy between these models in a similar task.

III. Dataset and Features

The FER2013 dataset is hosted by Kaggle, and was the subject of a machine learning competition in 2013 [7]. It includes approximately thirty-six thousand 48x48 grayscale photographs of human faces, each labelled with a particular emotion. Of the photographs provided, the training set contains 28,709 examples, and the validation and test sets each contain 3,589 examples. The images contain little to no background and include male and female models with various hairstyles, skin colors, and facial accessories (glasses, piercings, etc.). This study aims to find an efficient and accurate classifier to group



(a) Facial expressions



(b) Distribution of expressions

Fig. 1: (a) Seven different facial expressions. From left to right, top to bottom: (0) Angry, (1) Disgust, (2) Fear, (3) Happy, (4) Sad, (5) Surprise, (6) Neutral. (b) Distribution of emotions in the training set.

each image into one of seven emotions. Fig. 1(a) shows one example for each class.

A notable feature of this dataset is the distribution of emotions. As seen in Fig. 1(b), the number of "happy" expressions in the training set far exceeds that of "disgust". This class imbalance is taken into account in a couple of the models in this study by allocating appropriate class weights. Preprocessing is done on the inputs in the form of feature extraction in the case of non-neural network models and interpolation and expansion to multiple channels in CNNs. In the neural net architectures, various normalization methods are explored, including batch and dropout, and additional data augmentation is performed via random image transformations.

IV. Methods

A. *Histogram of Oriented Gradients (HOG)*

In addition to raw pixel values, the HOG equivalent of each image is used for feature extraction. This technique identifies edges in a photo by locating areas with high changes in color gradient. The resulting image output highlights these lines in whites and light grays while leaving the rest of the pixels dark gray or black.

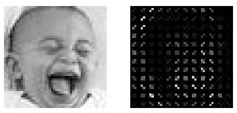


Fig. 2: Original image (left), HOG image (right).

B. *Logistic (Softmax) Regression*

Logistic, or softmax, regression is selected as the most basic algorithm to compare with the designed neural networks. One of the most widely used classification techniques, it operates as a single-layer network by assigning linear weights and an intercept term to the input values. Though the algorithm can be configured to support various kernels, only the standard linear kernel is included for this study.

While single-variable logistic regression has the sigmoid as its activation function, multi-class logistic regression uses the softmax, as defined below. This function estimates the probability that the input belongs to each of the output classes, and the algorithm then predicts the class with the highest probability.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{i=1}^{k} e^{z_i}} \tag{1}$$

Regularization in the logistic regression is explored as well, using Scikit-learn's C parameter at values 0.1, 0.25, 0.5, and 1.0 [8]. This adds a penalty term to the loss function of the form

$$\ell(\theta) = \text{Log Likelihood} - \frac{1}{C}\|\theta\|_1 \tag{2}$$

to discourage overfitting of the data.

C. *Support Vector Classifier (SVC)*

The SVC algorithm provides a more sophisticated competitor for the deep neural networks. SVC contains only a single layer but optimizes the geometric margin of its decision boundary as opposed to the function margin (as is done in softmax regression). The loss formula which the classifier attempts to minimize is as follows:

$$\min_{w,b,\gamma} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \epsilon_i \quad \text{such that} \tag{3}$$

$$y^{(i)}(w^T x^{(i)} + b) \geq 1 \quad i = 1, \ldots, m \tag{4}$$

The radial basis function (RBF) and polynomial kernels are trained in addition to the standard linear kernel to explore relations in higher dimensions. Results for polynomial kernels of degree two and three are included, and C parameter values are swept again from 0.1 to 1.0 to reduce overfitting. ϵ_i in the loss formula represents the margin of an incorrect prediction for input i. Tuning this parameter controls the trade-off between maximizing the geometric margin and classifying examples correctly.

D. *Convolutional Neural Network (CNN)*

A CNN is a deep learning model that is frequently applied to image recognition problems. It is a feedforward artificial neural network that consists of an input layer, output layer, and multiple hidden layers followed by a number of FC layers. The architecture of the hidden layers constitute the heart of a CNN model. Possible hidden layers include convolution, pooling, activation, and normalization layers. Training of a neural network involves parameter initialization (random, algorithmic such as Xavier initialization [9], or using pre-trained weights), evaluation of the loss function, and optimization. The CNN models developed in this study use the categorical cross-entropy loss, also known as the softmax loss, for multi-class classification:

$$CE = -\sum_{i}^{C} t_i \log\left(f(s)_i\right), \tag{5}$$

$$\text{where } f(s)_i = \frac{e^{s_i}}{\sum_{j}^{C} e^{s_j}} \tag{6}$$

The weights in each layer are updated per time step through SGD backpropagation.

E. *Residual Network (ResNet)*

ResNet is a deep learning algorithm introduced by the 2015 ILSVRC winner [5], with a concept known as identity shortcut connection at its core. Rather than hoping that each few stacked layers in a CNN directly fit a desired underlying mapping, ResNet explicitly fits them to a residual mapping, as shown in Fig. 3. The identity shortcut connection "skips" over the stacked layers, performing an identity mapping whose output is added to $F(x)$. This residual mapping proves easier to optimize than the original unreferenced mapping due to better diversity in its backpropagation paths and deals with the vanishing gradient problem prevalent in deep learning networks. The ResNet model thus enjoys improved accuracy from greatly increased depth while adding neither extra parameters nor computational complexity.

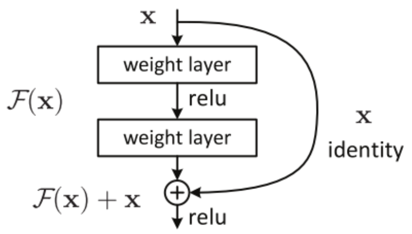

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 3: A ResNet block using identity shortcut connection.

V. Experiments, Results, and Discussion

Accuracy, measured by (# correct predictions / # total predictions), was the primary metric in determining the performance of a model. Other considerations included the amount of under- or overfitting between the training set accuracy and validation/test set accuracy.

A. *Raw Pixel Values (Non-Neural Networks)*

Both non-neural network models performed relatively well with raw pixels. SVC with RBF kernel established itself as the top performer with 41.5% test accuracy, while the logistic, SVC linear, and SVC poly models peaked at 34.8, 30.1, and 34.9%, respectively. All models experienced overfitting on the raw pixel data, likely due to the pairing of high-dimensional data with relatively simplistic, single-layer algorithms.

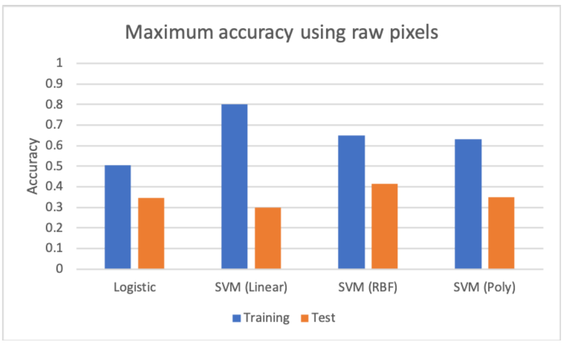
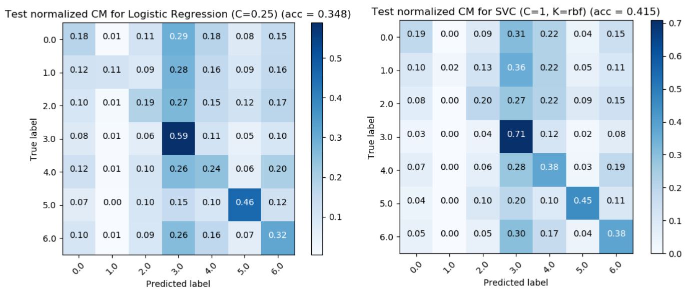


Fig. 4: Maximum accuracy with raw pixels in non-neural networks.

In order to better understand the models' behavior, confusion matrices were analyzed for each run. Fig. 5 reveals the simpler models' tendency to predict "happy" on most images. This stems from the class imbalance in our image dataset, which skews overwhelmingly toward this label. Though these models predict incorrectly approximately 65% of the time, the distribution of guesses is surprisingly even.

When tuning the models' C parameter (the regularization coefficient), neither the logistic regression nor the SVC linear model displayed notable changes in performance. Both the SVC RBF and SVC poly models reacted proportionally to the parameter, however, degrading in training and test accuracy as it lowered and improving as it increased. A quick analysis of their confusion matrices showed that as the penalty for misclassification dropped, the models predicted "Happy" for almost 100% of inputs.



(a) Logistic

(b) SVC RBF

Fig. 5: Confusion matrices using raw pixels.

B. *HOG (Non-Neural Networks)*

Models varied in performance with the HOG images as compared to the raw images. The SVC RBF model appeared unaffected, achieving almost the same accuracy of 41.5% on the test set. Both the logistic regression and the SVC linear model performed better with feature extraction over pixel representation, improving to 40.2 and 38.0%, respectively (increases of approximately five and eight points). The SVC poly model encountered difficulties, however, dropping over nine points to a 25.6% test accuracy.

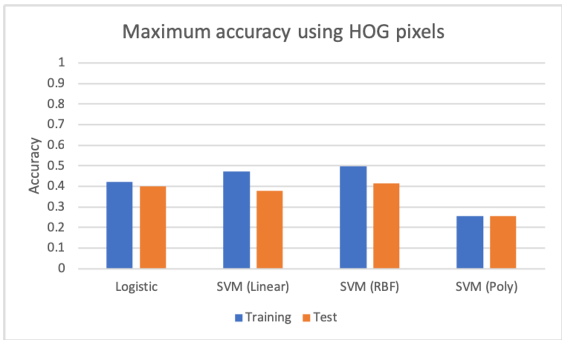


Fig. 6: Maximum accuracy with HOG pixels in non-neural networks.

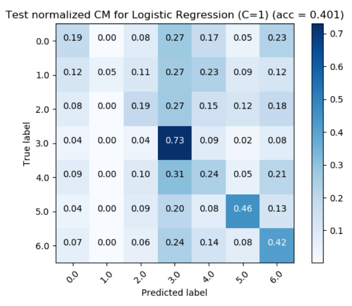
The overfitting of non-neural networks from the raw dataset disappeared with the HOG data, as all models displayed small margins between training and test set performance. This may have stemmed from the HOG representation's tendency to dramatically decrease the amount of information in the photos at low resolutions. While the raw images provided immense amounts of (arguably noisy) data that could induce overfitting on the training set, the HOG images ostensibly reduced those features to only those that matter.

Examining again the models' confusion matrices, is was found that the logistic model distributed its guesses more broadly and more aptly than before, while the SVC RBF model's predictions matched its original ones almost exactly.

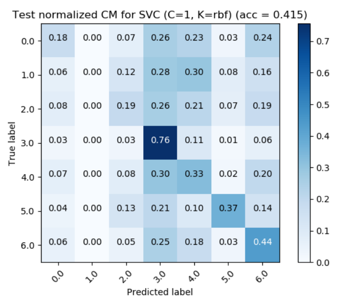
As for the hyperparameter C, the SVC poly model was unaffected by changes to this penalty term when using HOG pixels as inputs. The RBF kernel displayed similar degradation as before when the rate was lowered, a discrepancy that may be due to the difference in kernel dimensions.

C. *Baseline CNN Model*

A baseline CNN was built with (1) a [5×5] $\times$ 64 convolution layer followed by [3×3] max pooling with strides of 2, (2) a [5×5] $\times$ 64 convolution layer followed by [3×3] max pooling with strides of 2, (3) a [4×4] $\times$ 128 convolution layer, (4) an FC layer with 3072 neurons, and (5) a softmax output layer with seven neurons, one for each of the seven possible



(a) Logistic



(b) SVC RBF

Fig. 7: Confusion matrices using HOG pixels.

TABLE I: Confusion matrix for baseline CNN model, applied on test data

		True Values							
		0	1	2	3	4	5	6	Total
Predicted Values	0	**96**	13	31	24	37	9	22	232
	1	0	**0**	0	0	0	0	0	0
	2	87	8	**155**	62	86	82	63	543
	3	118	12	102	**690**	133	48	124	1227
	4	85	7	88	46	**178**	24	61	489
	5	11	1	56	10	13	**209**	11	311
	6	94	14	96	47	147	44	**345**	787
	Total	491	55	528	879	594	416	626	3589

facial expressions (Fig. 8). Each layer used Rectified Linear Unit (ReLU) as the activation function, and the algorithm was trained with a batch size of 64 over 20 epochs. The Adam optimizer with a learning rate of 1e-5 was used to minimize the categorical cross-entropy loss of the model. Training and validation error and loss were recorded for each epoch during the model compilation process.

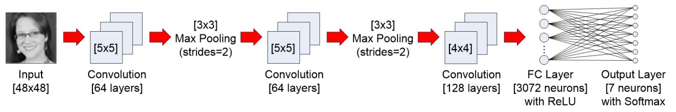

Input [48x48]
[5x5]
Convolution [64 layers]
[3x3] Max Pooling (strides=2)
[5x5]
Convolution [64 layers]
[3x3] Max Pooling (strides=2)
[4x4]
Convolution [128 layers]
FC Layer [3072 neurons] with ReLU
Output Layer [7 neurons] with Softmax

Fig. 8: Baseline CNN model architecture.

The accuracy of the baseline CNN model on the test set was 46.6%, with a loss of 1.41. The confusion matrix is given in Table I. Fig. 9 displays the accuracy and loss of the model on the training and validation sets over 20 epochs. It is immediately obvious that particular improvements could be made to improve this model. To begin, overfitting is virtually non-existent. This may be due to the learning rate of 1e-5 being too small, thus forcing the model to train for a longer time to obtain reasonable accuracy. Furthermore, it is clear from the confusion matrix that this model suffers from class imbalance. The "disgust" expression, which has label "1", is in fact never predicted at all. Meanwhile, the "happy" expression, which has label "3", is over-predicted. Since the "disgust" expression has the least number of available data points, while "happy" has the most, and because the baseline model is a rather simple CNN, this result is not completely unexpected. Proposed layers of complexity worth analyzing include adding convolution layers, increasing or decreasing filter size, adding batch normalization, adding dropout layers, and adjusting the number of neurons in the FC layer. Experimental results are outlined in the following sections.

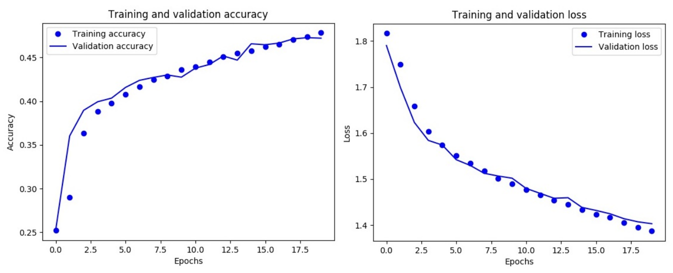


Fig. 9: Accuracy (left) and loss results (right) for baseline CNN model, plotted from 0 to 20 epochs.

D. *Fine-Tuning the CNN Baseline Model*

Adjustments were made to the baseline model, one at a time, in order to determine their effect on the CNN accuracy and loss. Since the baseline model exhibited underfitting, tests were performed to attempt to make a bias/variance trade-off and reduce the expected generalization error. Fig. 10 depicts the training and test error of these adjustments. It can be seen that increasing the learning rate from 1e-5 to 1e-4 reduced the amount of underfitting, significantly improving the training and test accuracy, as expected. Including batch normalization caused the model to overfit as well, with additional normalization layers contributing further to the discrepancy between training and test loss. Dropout, on the other hand, had little to no effect on either the amount of over- or underfit or model accuracy.

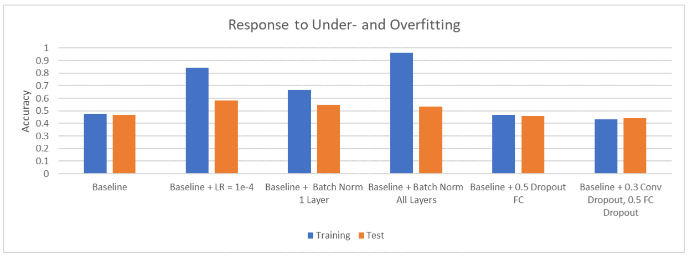


Fig. 10: Training and test accuracy based on adjustments made to learning rate and inclusion of batch normalization and dropout in the baseline model.

Additional general adjustments were made to the baseline model architecture (one by one) to determine their effect on the overall accuracy of the model. Fig. 11 shows the training and test accuracies with respect to varying convolution layer complexity. From the graph, it is not apparent that any significant improvements were made to the baseline model.

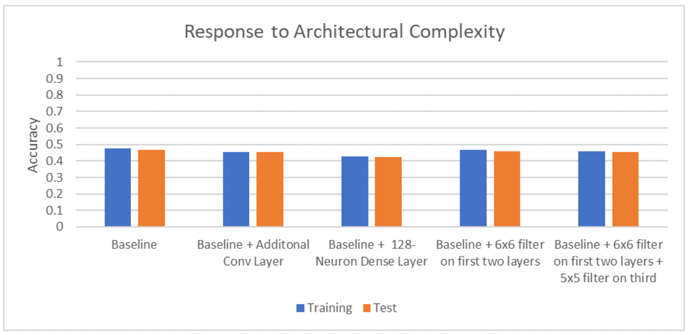


Fig. 11: Training and test accuracy based on adjustments made to number of convolution layers, FC layer neurons, and kernel size in the baseline model.

E. *Residual Network (ResNet50) Model*

ResNet50 from the Keras API was used to train and test on the FER2013 dataset [10]. GPU acceleration, which is supported in Keras with Tensorflow, helped to drastically decrease run times for very deep models [11]. Several optimizations, such as employing different batch sizes and breaking the training set into more manageable groups, were used as well. The

ResNet model was trained and tested on a personal computer with an Nvidia GTX 960 GPU. Even with these optimizations, computational runtimes still ranged from a few days to a week. Usage of Google Cloud services was considered, but not executed, due to the monetary cost of GPU support.

A baseline ResNet50 model from the Keras API was run using raw pixel images as input. First, only the FC layers, also known as top layers, were trained using the Adam optimizer with a learning rate of 1e-3 to minimize the categorical cross-entropy loss (all other layers were frozen). The number of epochs for this stage of training was set to three. After the top layers were trained, the convolution layers were fine-tuned. This time, the model was trained using the SGD optimizer, again minimizing for loss. The number of epochs for this stage of training was initially set to three as well.

The test accuracy of the baseline ResNet50 model was 40.5% with a loss of 1.54, and the confusion matrix is given in Table II. The accuracy obtained was much lower than anticipated. However, this was likely due to the low number of epochs in the baseline model. With each epoch taking about eight hours to run on a single GPU, a significant increase in training cycles was not feasible. In an alternative attempt to increase the test accuracy of the ResNet model, pre-trained weights from VGGFace2, a large-scale image dataset for facial recognition, were taken advantage of [12].

		True Values							
		0	1	2	3	4	5	6	Total
Predicted Values	0	**35**	5	27	3	9	23	9	111
	1	0	**0**	0	0	0	0	0	0
	2	30	5	**33**	26	21	27	21	163
	3	65	4	49	**574**	73	27	76	868
	4	133	16	161	155	**211**	39	153	868
	5	49	4	108	49	34	**252**	17	513
	6	179	21	150	72	246	48	**350**	1066
	Total	491	55	528	879	594	416	626	3589

TABLE II: Confusion matrix for baseline ResNet model, applied on test data

In the following experiment, the baseline model was run with the same parameters as described before, but with the weights initialized to those resulting from a previously trained model on VGGFace2. To comply with the pre-trained weights' corresponding image dataset, the original 48×48 images were re-sized to 197×197×3 input arrays. This was done by normalizing the images (subtracting off the mean of the pixel values), and then reshaping them to 197×197 images using nearest neighbor interpolation. Finally, the images were stacked three times to generate a 197×197×3 input array, simulating an RGB input of the VGGFace2 dataset. This array was then fed into the 3-Epoch-Top/3-Epoch-All model with pre-trained weights, producing a test accuracy of 63.3% with a loss of 1.08, and performing significantly better than the baseline model.

Further optimizations were attempted after producing the basic ResNet50 model, such as augmentation of the training set data. Augmentation is a powerful tool in image recognition models. For example, given an input image, randomly rotating it by a specified angle range, adjusting the brightness, mirroring it, or zooming in the image and adding it to the dataset would produce a new labelled data point on which the model could train. While humans can easily detect that a transformed image is essentially the same as the original, the pixel values that correspond to them differ significantly, adding diversity to the input data and increasing the robustness of the neural network. Fig. 12 shows examples of such image transformations.

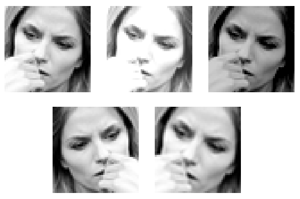


Fig. 12: From left, right, top, down: original, brighter, darker, horizontally flipped, and rotated image, all corresponding to the "sad" emotion.

The augmentations (1) rotational range from zero to ten degrees, (2) brightness range from zero to ten percent, and (3) horizontal reflection were added to the existing ResNet50 model. Zoom range was purposely excluded since the original input images already have little to no background and include only a person's face. The resulting model was then trained with three epochs for the top layers and three epochs for all layers. This achieved an accuracy of 66.1% on the test set, a nearly three point increase over the previous model. The confusion matrix is given in Table III.

		True Values							
		0	1	2	3	4	5	6	Total
Predicted Values	0	**285**	19	84	14	47	15	26	490
	1	6	**23**	2	0	2	0	1	34
	2	30	0	**165**	3	33	28	11	270
	3	34	5	28	**794**	45	31	41	978
	4	52	3	112	18	**328**	4	87	604
	5	14	2	65	14	4	**320**	3	422
	6	70	3	72	36	135	18	**457**	791
	Total	491	55	528	879	594	416	626	3589

TABLE III: Confusion matrix for ResNet model including VGGFace2 pre-trained weights and image augmentation, applied on test data

VI. Conclusion and Future Work

As expected, the neural network architectures produced significantly better accuracy than the simpler alternatives, outperforming them by more than twenty points. Logistic regression and the SVC with RBF kernel still achieved greater than 40% test accuracy, a decent margin above the naive solution of always predicting ”happy” (25% test accuracy).

Future work for non-neural networks in this study would include the implementation of additional feature extraction algorithms, such as Scale-Invariant Feature Transform (SIFT) or KAZE, a patent-free alternative. Evaluation of these feature algorithms mixed and matched with the investigated classification algorithms would be of interest. For logistic regression, class weighting and tuning the threshold for managing classification of different emotions would improve the observed class imbalance. Aditionally, the relative sensitivities of the SVC kernels to the hyperparameter C merit investigation.

A potential innovation regarding CNNs is to use feature extraction from HOG in conjunction with that of the convolution layers, feeding the result into one or more FC layers. Research is currently being done in this area, and the results are optimistic. On the other hand, the developed ResNet model could be trained for a longer number of epochs to achieve a potentially higher accuracy. Access to better hardware, such as high-performance supercomputing GPUs, would be extremely valuable in future analyses. Finally, implementation of early stopping and other regularization techniques in the optimizer could allow for faster learning rates without overfitting, reducing the required computation time.

References

[1] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems (pp. 1097-1105).

[2] CUDA GeForce, Nvidia, Santa Clara, CA, USA (2018). https://www.geforce.com/hardware/technology/cuda.

[3] Zeiler, M. D., and Fergus, R. (2014). Visualizing and understanding convolutional networks. In European Conference on Computer Vision (pp. 818-833).

[4] Simonyan, K., and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[5] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 770-778).

[6] Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A. (2014). Learning deep features for scene recognition using places database. In Advances in Neural Information Processing Systems (pp. 487-495).

[7] Kaggle, Google, Alphabet Inc. (2013). Challenges in representation learning: facial expression recognition challenge. https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge.

[8] Cournapeau, D. (2018). Scikit-learn, BSD License. https://scikit-learn.org/stable.

[9] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (pp. 249-256).

[10] Keras, MIT. (2018). https://keras.io.

[11] TensorFlow, Apache. (2018). https://www.tensorflow.org.

[12] Cao, Q., Shen, L., Xie, W., Parkhi, O. M., and Zisserman, A. (2018). VGGFace2: a dataset for recognising faces across pose and age. In IEEE Conference on Automatic Face and Gesture Recognition (pp. 1-8).

[13] Fan, R. E., Chang, K. W., Hsieh, C. J., Wang, X. R., and Lin, C. J. (2008). LIBLINEAR: A library for large linear classification. In Journal of Machine Learning Research (pp. 1871-1874).