Facial Expression Recognition Using Convolutional Neural Networks

Peibin Rui
prui@wisc.edu

Nolan Toole
ntoole@wisc.edu

Nilay Varshney
nvarshney2@wisc.edu

Abstract

Recognizing a person's emotions based on their facial expressions is an important part of life. It has estimated that over half of human communication is represented by facial expression, which leads to the importance of training a neural network to recognize what emotions facial expressions are linked with. In this project, we look at the performance of three different convolutional neural networks, AlexNet, VGG-19, and ResNet, to see which would be best for this task. In order to make sure the three can be compared, as much was kept the same between them as possible. This led to using the same training and test sets, as well as the same optimizer. What we found is that, while all three have some drawbacks, the ResNet that was implemented had the highest accuracy, with AlexNet second and VGG-19 third. However, when looking at how much time it took for training each model, it was the opposite finding, with VGG-19 taking the least amount of time and ResNet taking the largest amount of time.

Introduction

Human-Computer Interaction (HCI) is considered to be one of the fastest growing fields in technology today. As a result, systems that rely on HCI, like human-robot interaction systems and healthcare systems, have rapidly developed over the past several years. One area in which HCI can be improved is taking into account people's emotions. A large portion of human communication, 55 percent to be precise, is represented by facial expression. Therefore, in order for HCI systems to accurately determine people's emotions, it is important that they incorporate a well-performing Facial Expression Recognition (FER) model. [3]

FER is a widely studied topic in computer vision. Numerous papers that approach the topic have been published, and there are many databases of people's facial expressions that support these papers. It is commonly accepted that there are six basic facial expressions: fear, sadness, anger, disgust, happiness, and surprise, and many studies include neutrality as a seventh basic expression. [5] Therefore, in this project, we seek to find machine learning models that can accurately predict the seven basic facial expressions based on images of people.

The purpose of finding such models is two-fold. There are many benefits of having HCI systems that can determine people's emotions. Robots can use people's emotions to interact with them in an appropriate manner. This is especially advantageous in healthcare systems that can use people's behavioral expressions to help them improve their mental and emotional state. FER systems can also be used to diagnose certain mental diseases such as anxiety and autism. Another application of FER systems is in virtual reality (VR) and augmented reality (AR). VR and AR systems use FER to communicate with people naturally. [3] In short, we can use our research to improve the way computers can interact with humans, thus benefiting humans.

We can also personally benefit from doing this project. Right now, we are students with limited knowledge of deep learning and computer vision. Conducting this research will help us apply the material that we are learning in our Deep Learning class, thus solidifying our knowledge and experience in the areas of programming, cloud computing, and model development.

Figure 1. Example of facial expression recognition. Image source: https://towardsdatascience.com/speech-emotion-recognition-with-convolution-neural-network-1e6bb7130ce3

Related Work

Facial expression recognition using convolutional neural networks has been a popular topic in recent years. Lots of research has been done using different techniques in terms of CNN architectures and other factors.

Christopher and Martin [2] reviewed the state of the art

in image-based facial expression recognition using CNNs and highlight algorithmic differences and their performance impact. On this basis, they identified existing bottlenecks and consequently directions for advancing this research field. Furthermore, they demonstrate that overcoming one of these bottlenecks - the comparatively basic architectures of the CNNs utilized in this field - leads to a substantial performance increase. By forming an ensemble of modern deep CNNs, they obtained a FER2013 test accuracy of 75.2%, outperforming previous works without requiring auxiliary training data or face registration.

Yu and Zhang achieved state-of-the-art results in EmotiW in 2015 using CNNs to perform FER. They used an ensemble of CNNs with five convolutional layers each [4] . Among the insights from their paper was that randomly perturbing the input images yielded a 2-3% boost in accuracy. Specifically, Yu and Zhang applied transformations to the input images at train time. At test time, their model generated predictions for multiple perturbations of each test example and voted on the class label to produce a final answer. Also interesting is that they used stochastic pooling rather than max pooling because of its good performance on limited training data.

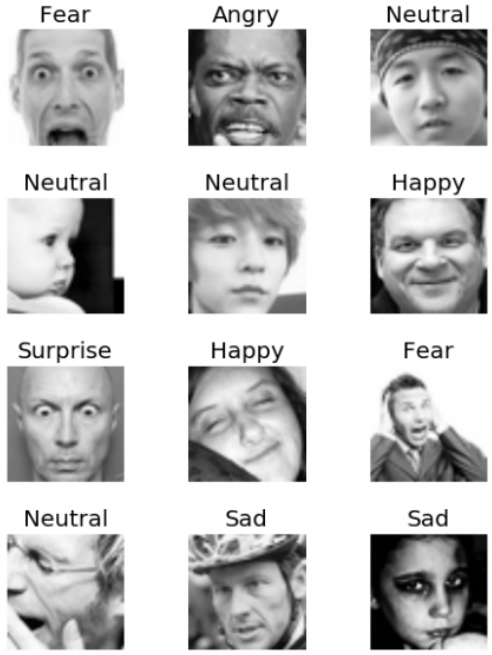

Figure 2. A sample of images from the data set, labeled with their corresponding emotions.

Shima and Azar [1] developed their models in Torch and exploited Graphics Processing Unit (GPU) computation in order to expedite the training process. In addition to the networks performing based on raw pixel data, they employed a hybrid feature strategy by which they trained a novel CNN model with the combination of raw pixel data and Histogram of Oriented Gradients (HOG) features. To reduce the over-fitting of the models, they utilized different techniques including dropout and batch normalization in addition to L2 regularization. They applied cross validation to determine the optimal hyper-parameters and evaluated the performance of the developed models by looking at their training histories. They also present the visualization of different layers of a network to show what features of a face can be learned by CNN models.

Instead of re-implementing published networks, we decided to take the key insights from these papers and experiment with different networks. We performed Alexnet, VGG-19 and ResNet to compare the result from different architectures and plotted the confusion matrix of each architecture to have a more straightforward understanding of the performance of each model.

Proposed Method

For this project, we consider three convolutional neural network architectures: AlexNet, VGG-19, and ResNet. For each architecture, we train the model on our training set, then evaluate its performance on the test set. We compare the three architectures based on their accuracy scores on the test set. We consider ourselves successful if at least one of the models we consider achieves a test set accuracy of greater than 50%.

AlexNet

AlexNet is the name of a convolutional neural network which has had a large impact on the field of machine learning, specifically in the application of deep learning to machine vision. It famously won the 2012 ImageNet LSVRC-2012 competition by a large margin (15.3% VS 26.2% (second place) error rates). The network had a very similar architecture as LeNet by Yann LeCun et al but was deeper, with more filters per layer, and with stacked convolutional layers. It consisted of 11×11, 5×5,3×3, convolutions, max pooling, dropout, data augmentation, ReLU activations, SGD with momentum. It attached ReLU activations after every convolutional and fully-connected layer.

- **ReLU Nonlinearity** AlexNet uses Rectified Linear Units (ReLU) instead of the tanh function, which was standard at the time. ReLU's advantage is in training time; a CNN using ReLU was able to reach a 25% error on the CIFAR-10 dataset six times faster than a CNN using tanh
- **Multiple GPUs** Back in the day, GPUs were still rolling around with 3 gigabytes of memory (nowadays those kinds of memory would be rookie numbers). This was especially bad because the training set

had 1.2 million images. AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU. Not only does this mean that a bigger model can be trained, but it also cuts down on the training time

- **Overlapping Pooling** CNNs traditionally "pool" outputs of neighboring groups of neurons with no overlapping. However, when the authors introduced overlap, they saw a reduction in error by about 0.5% and found that models with overlapping pooling generally find it harder to overfit

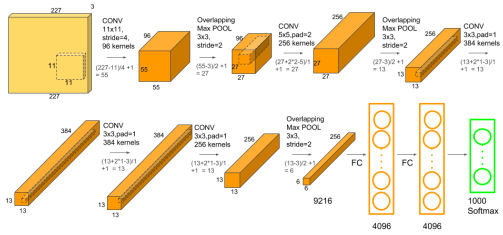

Figure 3. An illustration of how AlexNet works. Image source:https://neurohive.io/en/popular-networks/alexnet-imagenet-classification-with-deep-convolutional-neural-networks/

VGG

While previous derivatives of AlexNet focused on smaller window sizes and strides in the first convolutional layer, VGG addresses another very important aspect of CNNs: depth

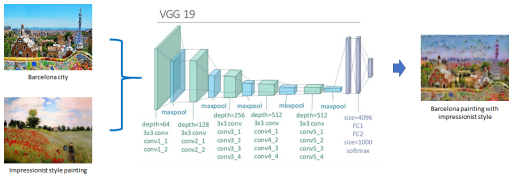

Figure 4. An illustration of how VGG works. Image source: `https://github.com/amenglong/art_generation_cnn`

- **Input** VGG takes in a 224x224 pixel RGB image. For the ImageNet competition, the authors cropped out the center 224x224 patch in each image to keep the input image size consistent
- **Convolutional Layers** The convolutional layers in VGG use a very small receptive field (3x3, the smallest possible size that still captures left/right and up/down). There are also 1x1 convolution filters which act as a linear transformation of the input, which is followed by a ReLU unit. The convolution stride is fixed to 1 pixel so that the spatial resolution is preserved after convolution
- **Fully-Connected Layers** VGG has three fully-connected layers: the first two have 4096 channels each and the third has 1000 channels, 1 for each class.
- **Hidden Layers** All of VGG's hidden layers use ReLU (a huge innovation from AlexNet that cut training time). VGG does not generally use Local Response Normalization (LRN), as LRN increases memory consumption and training time with no particular increase in accuracy.

The Difference. VGG, while based off of AlexNet, has several differences that separates it from other competing models:

- Instead of using large receptive fields like AlexNet (11x11 with a stride of 4), VGG uses very small receptive fields (3x3 with a stride of 1). Because there are now three ReLU units instead of just one, the decision function is more discriminative. There are also fewer parameters (27 times the number of channels instead of AlexNet's 49 times the number of channels).
- VGG incorporates 1x1 convolutional layers to make the decision function more non-linear without changing the receptive fields.
- The small-size convolution filters allows VGG to have a large number of weight layers; of course, more layers leads to improved performance. This isn't an uncommon feature, though. GoogLeNet, another model that uses deep CNNs and small convolution filters, was also showed up in the 2014 ImageNet competition.

ResNet

Plain Network The plain baselines (Fig. 5, middle) are mainly inspired by the philosophy of VGG nets (Fig. 5, left). The convolutional layers mostly have 3×3 filters and follow two simple design rules: (i) for the same output feature map size, the layers have the same number of filters; and (ii) if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer. It performs downsampling directly by convolutional layers that have a stride of 2. The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax. The total number of weighted layers is 34 in Fig. 5 (middle). It is worth noticing that the model has fewer filters and lower complexity than VGG nets (Fig. 5, left). The 34- layer baseline has 3.6 billion FLOPs (multiply-adds), which is only 18% of VGG-19 (19.6 billion FLOPs)

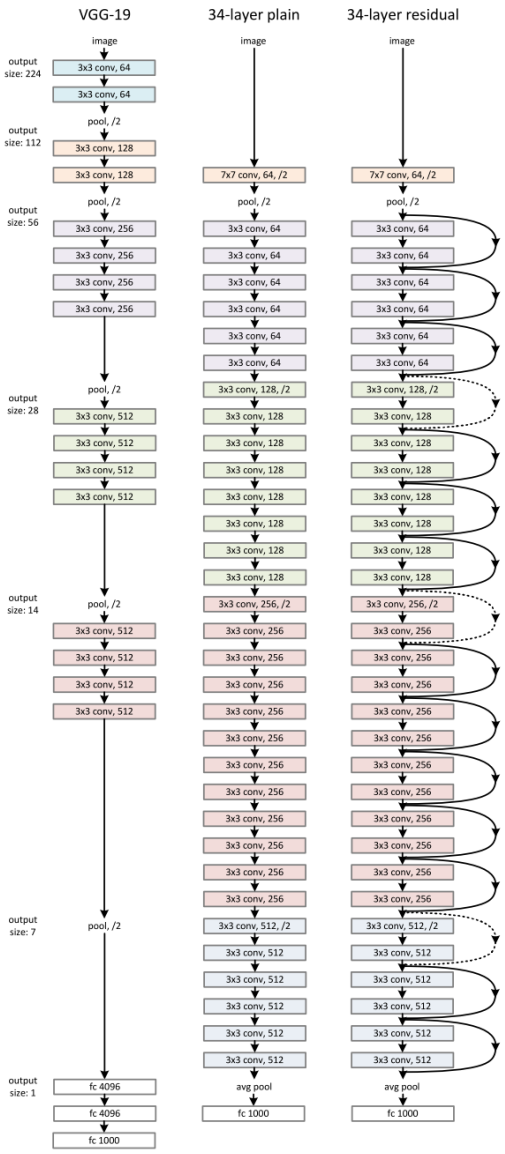

Figure 5. Example network architectures for ImageNet. **Left**: the VGG-19 model (19.6 billion FLOPs) as a reference. **Middle**: a plain network with 34 parameter layers (3.6 billion FLOPs). **Right**: a residual network with 34 parameter layers (3.6 billion FLOPs). The dotted shortcuts increase dimensions. Image source: https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035

Residual Network Based on the above plain network, we insert shortcut connections (Fig. 5, right) which turn the network into its counterpart residual version. The identity shortcuts can be directly used when the input and output are of the same dimensions (solid line shortcuts in Fig. 5). When the dimensions increase (dotted line shortcuts in Fig. 5), we consider two options: (A) The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter; (B) The projection shortcut is used to match dimensions (done by 1×1 convolutions). For both options, when the shortcuts go across feature maps of two sizes, they are performed with a stride of 2.

Accuracy

$$\text{Accuracy} = \frac{\sum_{i=1}^{n} I(\hat{y}_i = y_i)}{n}$$

n = Number of observations in test set
$\hat{y}_i$ = Predicted value of i^{th} observation of test set
y_i = True value of i^{th} observation of test set

It is important to note that due to the constraints imposed by our data set, we modified the three architectures. We discuss this further in the next section.

Experiment

The data set used in this project is the FER 2013 data set. This data set was used in the Kaggle competition "Challenges in Representation Learning: Facial Expression Recognition Challenge." The training set consists of 28,709 examples, and the public test set consists of 3,589 examples. The task is to categorize each face based on the emotion shown in the facial expression into one of seven categories (0 = Angry, 1 = Disgust, 2 = Fear, 3 = Happy, 4 = Sad, 5 = Surprise, 6 = Neutral). The data set can be found here: https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data

It is important to note that the FER 2013 data set is freely available. Many FER databases, like the Extended Cohn-Kadane database and the Radboud Faces database, require users to seek permission from their owners. Using a freely available data set allows us to independently conduct our analysis, instead of having to rely on database owners. It is also not guaranteed that the owners of the databases will allow us to use their databases for the purpose of our project.

The training set and test set contain 2,304 pixels per row, corresponding to 48 x 48 gray scale images. Because AlexNet, VGG-19, and ResNet all expect 224 x 224 RGB images, we needed to modify the architectures and/or the observations in the training and test sets so that the convolutional layers and max pooling layers would not shrink the inputs too much.

The first convolutional layer of the original AlexNet architecture expects an input with three channels. To accommodate for the fact that we are working with gray scale images, we change this layer so that it expects an input with only one channel. Furthermore, we noticed that if we keep the max pooling layers, the input shrinks to 0 x 0 before reaching the fully connected layers. Therefore, we remove the three max pooling layers.

For the VGG-19 architecture, it also requires 3 color channels. In order to meet the requirement, we made two

identical deep copies of the grey scale images and use them as the second and third color channels.

For the ResNet architecture, the main change made was to make stride a variable in the residual blocks, allowing for more control of when the image dimensions will be halved.

We use Python through Jupyter Notebooks and Google Colab for model training and evaluation, with PyTorch being the main Python package used. Additional functions developed by Sebastian Raschka were used in this project for computing and showing the confusion matrix for all models, which can be found on his GitHub, at https://github.com/rasbt/stat453-deep-learning-ss21. Each model is trained for 30 epochs with minibatches of size 64, and we use the Adam optimizer with an initial learning rate of 0.0003, and exponential decay rates of 0.9 and 0.999. We use a random seed of 123.

Results and Discussion

The AlexNet model was able to achieve a training set accuracy of 98.67% after 30 epochs. More importantly, it was able to achieve a test set accuracy of 51.83% on the test set, which means that the model was able to correctly guess the facial expression of 51.83% of the images in the test set.

The confusion matrix shown in Figure 6 shows the number of images with facial expression a that were predicted to have facial expression b for each of the 49 combinations of a and b. The figure reveals that the AlexNet model was able to correctly guess the facial expression for 46.47% of the images with an angry facial expression, 39.29% of the images with a facial expression of disgust, 36.69% of the images with a fearful facial expression, 68.16% of the images with a happy facial expression, 46.09% of the images with a sad facial expression accuracy, 63.61% of the images with a surprised facial expression, and 43.33% of the images with a neutral facial expression.

As of the training process of VGG-19, we achieved an accuracy of 94.31% after 30 epochs on the training set and an accuracy of 45.31% on the test set. Figure 7 shows the confusion matrix of this model. For exact percentages on each emotion, VGG-19 is able to correctly guess the facial expression for 34.90% of the images with an angry facial expression, 25.00% of the images with a facial expression of disgust, 32.46% of the images with a fearful facial expression, 59.44% of the images with a happy facial expression, 39.20% of the images with a sad facial expression accuracy, 57.35% of the images with a surprised facial expression, and 43.16% of the images with a neutral facial expression.

When training the ResNet model, a training accuracy of 99.25% was found when training for 30 epochs. Of more importance is the performance on the test set, finding an accuracy of 59.40%, showing that this model can correctly predict emotion based off of facial expressions 59.40% of the time. Although this is the highest test set accuracy out of the models we used, the model also took by far the longest to train, taking over 45 minutes for the 30 epochs.

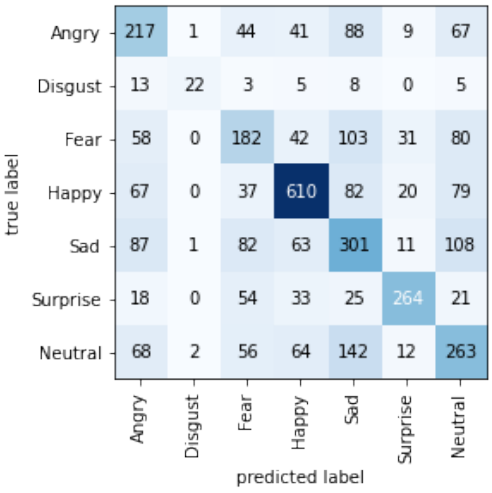

Figure 6. Test set confusion matrix for AlexNet

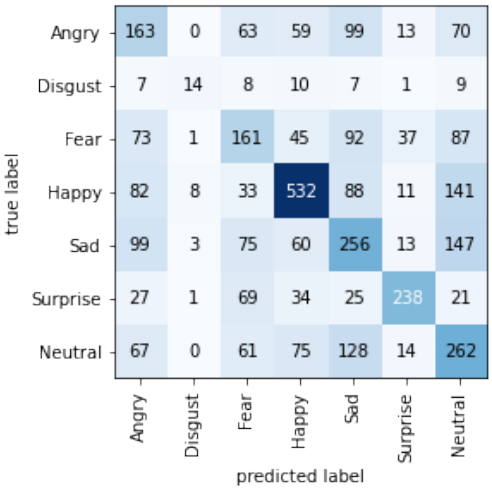

Figure 7. Test set confusion matrix for VGG-19

Figure 8 shows the confusion matrix from this model, showing similar results to both other models in that the highest accuracy is on predicting happy from the images. For exact percentages on each emotion, facial expressions that were angry were correctly guessed by the model 54.39% of the time, expressions of disgust 41.07% of the time, expressions of fear 32.86% of the time, expressions of happiness 81.45% of the time, expressions of sadness 47.47% of the time, expressions of surprise 74.22% of the time, and neutral expressions 57.12% of the time.

Out of the models used, there was a clear difference in test set accuracy for all three. The highest accuracy was the

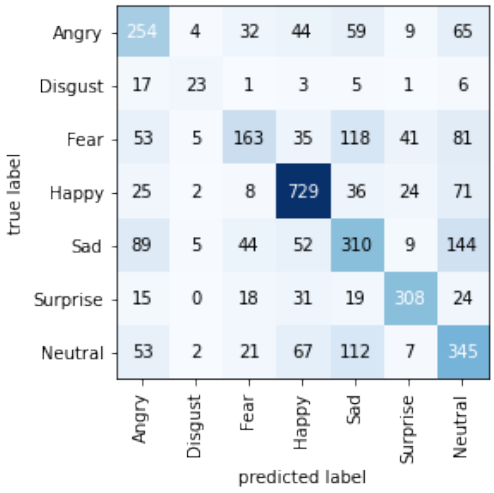

Figure 8. Test set confusion matrix for ResNet

ResNet model, achieving a test set accuracy of 59.40%, followed by AlexNet with an accuracy of 51.83%, and VGG-19 with an accuracy of 45.31%. Something noteworthy for each model is the run-time for training the 30 epochs, as the ResNet model took over 45 minutes to train, with AlexNet and VGG-19 taking much less time, at about 15 and 25 minutes respectively. Each model was rather similar in which emotion was guessed correctly the most and least. Each model had its highest accuracy on images with a happy expression, as AlexNet had 68.16%, VGG-19 had 59.44%, and ResNet had 81.45% accuracy.

Conclusions

The goals of this project were completed, as we were successful in implementing three different convolutional neural networks for facial expression recognition. Since all three were correctly implemented, that allowed us to also be able to compare the accuracy and speed of AlexNet, VGG-19, and ResNet. This comparison showed us that the most accurate convolutional architecture for facial expression recognition of the three is ResNet, followed by AlexNet then VGG-19. However, in terms of amount of time for each architecture took on training, ResNet took by far the most, with AlexNet taking the least amount of time, nearly three times faster than ResNet. Each model was also similar in which expressions were commonly correct from the model, with expressions of happiness having the highest accuracy in all three models.

However, none of the convolutional neural networks implemented in this project had a significantly high accuracy. With the highest accuracy being 59.40%, it is much lower than the 75.2% that was found by Christopher and Martin [2]. This shows us that further improvements can be made in our implementations of these networks. Some possible improvements could also be made in the dataset used, as the images in FER 2013 are 48x48 and grayscale. Using larger and in color images for the data could lead to higher accuracy from what is already implemented.

Acknowledgements

The data used in this project was obtained from the FER 2013 data set used in the Kaggle competition ”Challenges in Representation Learning: Facial Expression Recognition Challenge”. Helper functions made by Sebastian Raschka used for making and displaying confusion matrices. Google Colab used for writing and running the code.

Contributions

Nolan Toole was the main leader for this group. He also evaluated the performance of the ResNet architecture on the FER 2013 dataset. Peibin Rui evaluated the performance of the VGG-19 architecture on the FER 2013 dataset, and Nilay Varshney evaluated the performance of the AlexNet architecture on the FER 2013 dataset. Toole, Rui, and Varshney each contributed equally to the presentation and report.

References

[1] S. Alizadeh and A. Fazel. Convolutional neural networks for facial expression recognition. *CoRR*, abs/1704.06756, 2017.

[2] C. Pramerdorfer and M. Kampel. Facial expression recognition using convolutional neural networks: State of the art. *CoRR*, abs/1612.02903, 2016.

[3] N. Samadiani, G. Huang, B. Cai, W. Luo, C.-H. Chi, Y. Xiang, and J. He. A review on automatic facial expression recognition systems assisted by multimodal sensor data. *Sensors*, 19(8), 2019.

[4] Z. Yu and C. Zhang. Image based static facial expression recognition with multiple deep network learning. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI ’15, page 435–442, New York, NY, USA, 2015. Association for Computing Machinery.

[5] M. Zavarez, R. Berriel, and T. Oliveira-Santos. Cross-database facial expression recognition based on fine-tuned deep convolutional network. pages 405–412, 10 2017.