Jackson Fries
2/7/2024
ECE 442
Image Classification

Section 1: Summary of Image Classification Methods

One of the biggest uses for machine learning is image classification. This is when a machine learning model is handed training data consisting of different pes of images and the model is then tasked to correctly identify each picture. These pictures can consist of almost anything but a few examples would be identifying handwritten text with an image database such as MNIST. This can be useful for a whole variety of different applications. Some of the most important ones are medical imaging, facial recognition, and autonomous driving. When looking into the different models for image classification there are only a few real contenders. These two models are CNN's(convolutional neural networks) and ANN's(artificial neural networks).ANN's can also be called Deep Neural Networks. There are also support vector machines (SVM's) in many of the studies referenced, however they are usually fairly far behind deep learning in terms of time complexity and accuracy. Therefore they will not be referenced much going on. These two image classification models make up the bulk of models that are commonly used in image classification with deep learning. CNN's making up the majority of image classification methods as they are usually significantly more accurate. However, CNN's are not the end all method where image classification is concerned as even though for most image classification methods CNN's are king, when the application strays further away into more pattern recognition and in smaller models they become less ideal. This is due to the intense requirements that CNN's call for computation wise. Another weakness of CNN's is that they typically require more training data and longer training times. This leaves room open for less complex image classification methods to edge a small niche. However, over time greater computational power and larger data sets are becoming less of a worry as it becomes easier to collect data and computers become more powerful. Another important factor that goes into the speedup of CNN's is the continued research that goes into speeding up the algorithms that power them.

This can be seen in an experiment conducted by K.T. Islam, R. G. Raj, and A. Al-Murad where they tested various different image classification methods on different faces. The dataset they used here is very important as they fed the NN's with 20 different faces with ten photos of each, this ended up with a total of only two hundred images. This is an extremely small sample set which should give ANN's an advantage over CNN's as CNN's tend to need more sample photos, however the difference between the two was marginal at best. With different color filters applied to the image to also see which type of image trained the models the best, they found that out of the three different image types they tested CNN's two of the three had the exact same accuracy and was only marginally slower than the ANN. The only test in this experiment that resulted in CNN's being less optimal was when they used the BoW(bag of words) filter, but even

then it was only two percent less accurate. This data reveals that even though in some certain aspects ANN's can perform slightly better than CNN's, CNN's at a base level are better or the same as ANN's even in its worse conditions. When looking at how CNN's essentially compress the images so that they are easier for the computer to classify.

Another experiment was done by D. Beohar and A. Rasool to see the difference between ANN's and CNN's. To do this they used the MNIST data set to classify handwritten characters. The results from this study were very clear, CNN's had the upper hand over ANN's. They found that over the course of ten epochs, the ANN had an error rate of 1.31% while the CNN had an error rate of .91%. They also found that while the CNN had a better accuracy rate it also took longer and used more resources in the CPU. They also said that this could be further optimized by also using a GPU. This data just further shows that CNN's are the superior image classification method in machine learning even though they can be quite computationally intensive.

Section 2: How CNN's and other NN's work

To understand how CNN's work it is essential to understand how ANN's and lower level machine learning processes work as CNN's has many parts that are made up out of these smaller level pieces. The information for this next section mainly comes from the paper written by Chen L et al. The basis of almost every neural network is the neuron or node. The neuron takes in the input from the layer before and essentially takes the dot product of all its inputs. It will also add a weight and a bias to this. It can then send this new value to the next layer in the neural network. The next level up from neurons is a Multilayer perceptron or MLP for short. An MLP is the most basic neural network which consists of usually only three layers, the input, the hidden layer, and the output. The input layer takes in the input data and starts to transmit that out the hidden layer. The hidden layer pretty much looks at the input data and depending on the weights and biases will look for certain patterns in the input. For example if you input a 9 into an MLP each pixel would go into each input neuron and from there it would be sent to every neuron in the hidden layer, in which a pattern such as edge recognition will be found and sent out to the output layer to see what it classified it as. What the designer sets as the weight and bias will influence what type of pattern is recognized in that layer. The next step up from this is an ANN in which there are usually multiple hidden layers to get better recognition. The thing that the CNN does differently while also building upon these basic structures is that it adds two new types of layers. The convolution layer and the pooling layer. The convolution layer is where the CNN gets its name and it is made up of multiple filters called kernels. These kernels are essentially little matrices that will parse over the inputted image and will be specialized into looking for specific types of patterns in the image by taking the dot product of the kernel itself and the section of the image it overlays. It will then store this output into a new matrix called the feature map. There are usually multiple different types of kernels in any given ML system. For more basic ones such

as letter classification using the MNIST database, there would be types such as edge detectors and ones that would classify specific common shapes like the loop of a lowercase g. Once the kernels have parsed over the input image and created feature maps, these feature maps are then sent onto the next unique layer in a CNN, the pooling layer. The pooling layer essentially takes these feature maps and downsamples them into a smaller size to reduce the overall complexity of the calculations further down the line. For example if the feature map is a 28x28 image the pooling layer would turn into a 14x14 size image. This would reduce the complexity greatly. The way it does this is by taking the most defining attributes in the feature map and getting rid of the rest, there are a couple different types of pooling such as max, min or avg. These different types determine what values are saved from the feature maps. For example, max pooling would take the highest value in a certain region of the map and translate that over to the downsized feature map, it essentially takes the highlights of each section. Min pooling would do the opposite, taking the lowest value in each section. These two types of layers can be repeated again and again to build up more complex shape recognition. The first layers in a network usually pick out rough shapes or objects such as lines and edges, and later layers look for things such as a triangle or swirl. Once the CNN is done with the pooling and the convolutional layers, the final feature map is then sent to the fully connected or FC layers. These FC layers are just the MLP discussed earlier, the only difference being that the input to these layers are the highly abstracted feature maps coming out from the convolution and pooling layers. The CNN as known today can be seen as almost an add on to ANN's as what it achieves is making a more easily classifiable and efficient image for the FC's to classify.

3. Comparison Table of different NN's

Facial Recognition IP (K.T Islam, et al 2017)

	CNN	ANN
Speed	21.84	15.58
Accuracy	99.50%	99.50%

MNIST Digit Classification(Kadam, Shivam, et al. 2020)

	CNN	ANN
Speed	3500 seconds	100 seconds

Accuracy	99.08%	98.29%

Vegetation Species(Hasan M, et al. 2019)

	CNN	ANN
Speed	NA	NA
Accuracy	99%	94%

4. Findings for CNN simulations

One of the main purposes of this paper is to give an estimate to the parameters and specs needed to implement our own CNN on the MNIST database to recognize handwritten digits. Some of the important aspects that need to be estimated are kernel size, pooling type, and amount of layers.

I. Estimated Kernel Size

In a study conducted 2020 by Shivam S. Kadam, Amol C. Adamuthe, and Ashwini B. Patil conducted a study on 5 different CNN architectures which included different amounts of layers and different kernel sizes. By the end of the study they found that for the MNIST dataset specifically anything past a 2x2 kernel was overkill. They found that there was almost a double in the training time required with only a marginal increase in accuracy from 99.03% to 99.29%. These tests had the same amount of convolution and pooling layers with the main difference being the kernel size. It is due to this inefficiency that I would recommend that for our simulations we would use a 2x2 kernel. However if training time was a non issue then I would recommend a 3x3 for the added accuracy.

II. Estimated Number of Convolution and Pooling Layers

From the study conducted by Shivam S. Kadam, Amol C. Adamuthe, and Ashwini B. Patil in which they conducted experiments using five different CNN architectures in which they adjusted the amount of convolution and pooling layers. Throughout from architectures two to four they increased the amount of convolutional layers by one each time. From architecture two to three in which it went from two to three convolution layers they found an increase in accuracy from 98.96% to 99.37% a very sizable increase. However when they increased to four convolution layers in architecture four the accuracy actually went down from 99.37% to 99.03%. Another major factor here is that by adding more convolution layers the time it took to train also

went up. Each step increases the time complexity by 100 seconds. With architecture 2 taking 3500 seconds and architecture 4 taking 3700 seconds. However, when making our own CNN it is likely that it will not be as optimized as most other CNN's are. This could lead to a massive increase in training time with each added layer. For this reason I would suggest 1 convolution layer and 1 pooling layer as it will be faster with good accuracy.

III. Best Pooling Method

When looking at the pooling method it is first important to realize how the MNIST database is set up. The MNIST database is composed of white digits on a black background, this type of image lends itself really well to max pooling as it is very obvious to see where the lightest part of the images are. Implementing our simulations with max pooling is the most optimal option.

Works Cited:

K. T. Islam, R. G. Raj and A. Al-Murad, "Performance of SVM, CNN, and ANN with BoW, HOG, and Image Pixels in Face Recognition," 2017 2nd International Conference on Electrical & Electronic Engineering (ICEEE), Rajshahi, Bangladesh, 2017, pp. 1-4, doi: 10.1109/CEEE.2017.8412925. keywords: {Feature extraction;Face;Support vector machines;Face recognition;IP networks;Artificial neural networks;Training},

D. Beohar and A. Rasool, "Handwritten Digit Recognition of MNIST dataset using Deep Learning state-of-the-art Artificial Neural Network (ANN) and Convolutional Neural Network (CNN)," 2021 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2021, pp. 542-548, doi: 10.1109/ESCI50559.2021.9396870. keywords: {Deep learning;Training;Handwriting recognition;Feature extraction;Central Processing Unit;Convolutional neural networks;Image classification;Handwritten digit recognition;Convolutional Neural Network (CNN);Deep learning;MNIST dataset;Epochs;Hidden Layers;Stochastic Gradient Descent;Backpropagation},

Kadam, Shivam S., et al. "CNN model for Image Classification on Mnist and fashion-mnist dataset." Journal of Scientific Research, vol. 64, no. 02, 2020, pp. 374–384, https://doi.org/10.37398/jsr.2020.640251.

Chen L, Li S, Bai Q, Yang J, Jiang S, Miao Y. Review of Image Classification Algorithms Based on Convolutional Neural Networks. Remote Sensing. 2021; 13(22):4712. https://doi.org/10.3390/rs13224712

Hasan, M., Ullah, S., Khan, M. J., and Khurshid, K.: COMPARATIVE ANALYSIS OF SVM, ANN AND CNN FOR CLASSIFYING VEGETATION SPECIES USING HYPERSPECTRAL THERMAL INFRARED DATA, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLII-2/W13, 1861–1868, https://doi.org/10.5194/isprs-archives-XLII-2-W13-1861-2019, 2019.