Analysis of Pre-trained CNN Models on Various Image Data Sets

Ian Rector

Abstract:

There are numerous different pre-built convolutional neural network architectures. I analyzed the LeNet5, VGG16, InceptionV3 and ResNet50 architectures on 4 different datasets: Cifar10, Cats vs. Dogs Redux: Kernel addition, Eurosat and Horses vs. Humans. Each of these datasets involve different kinds of comparison and 2 are multiclass and 2 are binary classification problems. The datasets also have different image sizes and numbers of images. I found that VGG16 and InnceptionV3 performed the best across all 4 datasets.

Introduction:

There are numerous different ways to structure a convolutional neural network and many kinds of pre-built architectures. Most pre-built architectures require images with RGB coloration, so the input shape is (x, x, 3) with x being the dimensions of the image. Some models, like the InceptionV3 also require images to be at least [75, 75]. I selected four different models: LeNet5, VGG16, InceptionV3 and ResNet5 since all of them are quite different in structure and so could provide unique insights on which models serve different image sets best. LeNet5 is a simple early architecture designed more for digit classification tasks and so will act like a control (Bangar 2022). ResNet50 on the other hand is quite complicated with InceptionV3 and VGG16 somewhere in between (Yan 2023). I selected four different datasets: Cifar10, Dogs vs. Cats Redux, Eurosat and Horses vs Humans. Dogs vs. Cats Redux was obtained through Kaggle and all of the other datasets are available from the tensorflow datasets package. I selected these datasets due to differing numbers of images, different sizes and different features. 2 of the datasets have a binary selection: Dogs vs. Cats and Horses vs. Humans. The Dogs vs. Cats dataset features more images and larger images than the Horses vs. Humans dataset, but dogs and cats have similar features making it harder to discern the two. The Horses vs. Humans dataset

uses computer generated images with subjects that look fairly different; however, the dataset was quite small containing only slightly over 1000 images for the train, test and validation datasets. We have already looked at the Cifar10 dataset, but it provides a multiclass problem with very different images. The Eurosat dataset is composed of geographical features in Europe. This makes it quite different from the other sets, but is also a multi-class problem. I initially wanted to try a relatively simple dataset like MNIST, but the models other than LeNet5 require RGB coloration. This likewise ruled out X-ray image datasets that I wanted to analyze due to the coloration issues.

Literature Review:

LeNet5 was one of the first CNN models introduced and represented a breakthrough in 1989 (Bangar 2022). However, it has been surpassed by many different architectures today as we will see in the **Results** section. ResNet50 is a much more advanced model that utilizes skip connections to help eliminate the vanishing gradient problem (Yan 2023). VGG16 is an architecture from an Oxford group that is useful for a variety of image classification tasks and is composed of 16 different convolution layers (Yan 2023). InceptionV3 was developed by Google and utilizes multiple parallel convolution layers of various sizes (Yan 2023). ResNet50, VGG16 and InceptionV3 were all trained on the ImageNet dataset (Yan 2023).

A similar study from Darapaneni et al. examined the performance of 18 different models on the Cifar-10 dataset. Cifar-10 is a very popular dataset for this sort of analysis. They found the ‘Squeeze and Excitation’ network was the most accurate with around 95% accuracy and out of the models I will be using InceptionV3 was the most accurate. Another similar study from Yalov-Handzel et al. examined the performance of 3 different architectures (EffiecientNetB3, ResNet50

and InceptionV3) on 10,000 archeological images. This study also finds the InceptionV3 model to perform the best although not that much better than ResNet50.

Methods:

As mentioned above, four different models were tested: LeNet5, VGG16, InceptionV3 and ResNet50. Each of these models were tested on the four datasets. The four basic model architectures are shown in appendix figures 1-3. The InceptionV3 architecture is not shown since when the InceptionV3 model is depicted it includes all layers. The VGG16 and ResNet50 architectures are easier to display since they are composed of their respective blocks followed by any penultimate layers and final output layers. To maintain consistency, the optimizer for all of the models was 'adam' and they were trained using the accuracy metric. All models for the Cifar10 dataset used 'categorical cross-entropy' for the loss function and all models for the Eurosat dataset used sparse categorical cross-entropy for the loss function. This had to do with how the labels were processed. For the other two binary datasets 'binary cross-entropy' was used as the loss function. The final output layer for the multiclass datasets had 10 nodes and used a 'softmax' activation function. For the Dogs vs. Cats dataset, I did something unconventional and the final layer for these models was a dense layer with 2 nodes using a 'softmax' activation function. I recognize that for binary classification the final layer typically only has 1 node and uses a 'sigmoid' activation function, which is what I did for the 'Horses vs. Humans' dataset. However, due to how the data was preprocessed the 1 node and 'sigmoid' activation functions were producing strange results that I will detail in the results section. I do not think that this unconventional final layer was a problem considering the great results that we see with that dataset with the InceptionV3 and VGG16 models.

Each model was trained using 10 epochs and a train test split of 80:20 was employed on the first 3 datasets. Furthermore, a validation set of 10% of the train set was created for the first three datasets. Due to the small size of the 'Horses vs. Humans' dataset the split was 80% train, 10% validation and 10% test.

The Dogs vs. Cats dataset was obtained through Kaggle (https://www.kaggle.com/competitions/dogs-vs-cats-redux-kernels-edition) and the other datasets are available through the tensorflow-datasets Python package. The LeNet5 model needed to be created in Python since a package does not exist for it since it is a simpler model than the other. The VGG16, InvceptionV3 and ResNet50 blocks were created using their respective Python packages. The Cifar10 and Eurosat datasets needed to be resized to [75,75] for the InceptionV3 model since that is the minimum input size it takes in. All models except for LeNet utilized 'imagenet' for the preset weights. LeNet5 does not have preset weights, so this did not apply to that model.

Results:

Results for the Cifar10 dataset were not great with none of the models achieving greater than 64% accuracy. The ResNet50 model did not even achieve 40% accuracy on the test set. This is what I found when I tested some of these models during the second assignment. None of the models were able to achieve the 70+% accuracy that is easily achieved with simpler CNN architectures. The Dogs vs. Cats dataset was a different story with the InceptionV3 model achieving 98% accuracy and the VGG16 model achieving 93% accuracy. ResNet performed better, but still only achieved 67% accuracy. This dataset did not work well for the LeNet5 model as it identified every image in the test set as a cat leading to sub 50% accuracy. As mentioned in the **Methods** section I created the output layer of these models with 2 nodes and used a 'softmax'

activation function, which is generally not the best practice for binary classification problems. However, when I used 1 node and a 'sigmoid' activation function the more complex models like VGG16 had test accuracies locked at 50% and the LeNet5 model still classified every image as either 1 of the 2 available classes, sometimes cats and sometimes dogs. Due to this I found that utilizing 2 nodes and a 'softmax' activation function in the final layer was the best option. It did not work well for LeNet5, but that model performed poorly on all the datasets and performed poorly whether I used a 'softmax' or 'sigmoid' activation function. This approach did lead to high accuracies for both the InceptionV3 and VGG16 models.

The models performed better on the Eurosat dataset than the Cifar10 dataset. This is interesting since the Euorsat dataset is geographic landforms, which look more similar to a human than the diverse images represented in the Cifar-10 dataset. However, there are discernible features such as blue representing some sort of water formation and different lines on the images representing different plots of land that would make these images more discernible to the models. This time the VGG16 model outperformed the IncepctionV3 model. The VGG16 model achieved an accuracy of 87% on the test set and the InceptionV3 model achieved an accuracy of 83%. Once again, the ResNet and LeNet5 models were dramatically worse achieving accuracies of 58% and 48% respectively.

The final dataset was much smaller than the other 3 (around 1000 total images as opposed to between 20 and 30 thousand for the other 3). However, horses and humans have discernible features, so the models were able to achieve very high accuracies even with the limited sample size. Like with the Dogs vs. Cats dataset, the LeNet5 dataset classified every image as a human, which resulted in sub 50% accuracy once again. The InceptionV3 model on

the other hand was able to achieve 100% accuracy on the test set. The other two models achieved greater than 93% accuracy.

In addition to accuracy, time can be a major restraint on which model someone chooses to use. The LeNet5 model was consistently the fastest followed by InceptionV3 and ResNet50. VGG16 was by far the slowest. InceptionV3 was 5 times faster than VGG16 and LeNet5 was almost 10 times faster on this dataset (**Figure 4**). Confusion matrices, plots of loss, plots of accuracy and other preprocessing of images are shown in the code, but I thought not to include these plots in the appendix due to potential clutter.

Conclusion:

For both binary classification problems, the LeNet5 model classified every image as the same class so was around 50% accurate. This model is from 1989, so was chosen more as a control, but even so I expected better results than what it was able to produce. It likely did not perform well with these colored datasets since it is much better suited for black and white digit classification tasks like MNIST (Bangar 2022). Its poor performance might have to do with a lack of dropout layers and use of the tanh activation function; however, this requires further examination. It is fascinating to see how far CNNs have advanced in the past 35 years since a typical CNN architecture that someone could create from scratch today is likely to dramatically outperform the LeNet5 model at least on classification tasks involving colored images.

The ResNet50 model also performed poorly on every dataset except the Horses vs. Humans dataset, but the features in these images were quite separable making this a fairly easy classification task despite the small size of the dataset (around 1000 images). The ResNet50 model is very elaborate and is probably better suited for these large, complicated datasets, but I

unfortunately lacked the time needed to do that for this assignment (Yan 2023). For the chosen datasets its performance was quite underwhelming. Overall, the InceptionV3 model performed the best with the VGG16 model also dramatically outperforming the other two models. Given how much faster the InceptionV3 model is compared to the VGG16 model, InceptionV3 seems to be the best choice if you are selecting a pre-built model. This confirms the results seen in both Darapaneni et al. as well as Yalov-Handzel et al. I should have also tested a model of my own design since it does seem like for the more basic image classification tasks that selecting a pre-trained model will not yield substantially better results than creating one's own model.

The paper from Darapaneni et al. was exhaustive in its use of models utilizing 18 different models whereas the Yalov-Handzel paper only utilized 3 different architectures. However, both papers only examined 1 dataset and did not use a cross-sectional approach like I did. Future experiments utilizing extensive models like in Darapaneni et al. as well as cross-sectional approaches should continue to be done to figure out the best use cases of these powerful models.

References:

Bangar, Siddesh. 2022. “LeNet5 Architecture Explained”. Medium. Last modified June 22, 2022. https://medium.com/@siddheshb008/lenet-5-architecture-explained-3b559cb2d52b

Darapaneni, Narayana, B. Krishnamurthy and Anwesh Reddy Paduri. 2020. "Convolution Neural Networks: A Comparative Study for Image Classification," *2020 IEEE 15th International Conference on Industrial and Information Systems (ICIIS)*: 327-332. doi: 10.1109/ICIIS51140.2020.9342667.

Yalov-Handzel, Sharon, Ido Cohen, and Yehudit Aperstein. 2024. ‘Comparative Analysis of CNN Architectures and Loss Functions on Age Estimation of Archaeological Artifacts’. *Journal of Computer Applications in Archaeology*, 7(1): 185–194. https://doi.org/10.5334/jcaa.136.

Yan, Kevin. 2023. “These are the 5 best pre-trained neural networks”. Medium. Last modified July 8, 2023. https://medium.com/@kyan7472/these-are-the-5-best-pre-trained-neural-networks-23798e61a043

Appendix:

Figure 1: LeNet model for Cifar10

```
Model: "sequential_28"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_243 (Conv2D)         (None, 32, 32, 6)         456

 average_pooling2d_54 (Aver  (None, 16, 16, 6)         0
 agePooling2D)

 conv2d_244 (Conv2D)         (None, 12, 12, 16)        2416

 average_pooling2d_55 (Aver  (None, 6, 6, 16)          0
 agePooling2D)

 conv2d_245 (Conv2D)         (None, 2, 2, 120)         48120

 flatten_27 (Flatten)        (None, 480)               0

 dense_72 (Dense)            (None, 84)                40404

 dense_73 (Dense)            (None, 10)                850

=================================================================
Total params: 92246 (360.34 KB)
Trainable params: 92246 (360.34 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 2: VGG16 model for Cifar10

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 1, 1, 512)         14714688

 flatten_3 (Flatten)         (None, 512)               0

 dense_10 (Dense)            (None, 512)               262656

 dense_11 (Dense)            (None, 10)                5130

=================================================================
Total params: 14982474 (57.15 MB)
Trainable params: 267786 (1.02 MB)
Non-trainable params: 14714688 (56.13 MB)
_________________________________________________________________
```

Figure 3: ResNet50 model for Cifar10

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 1, 1, 2048)        23587712

 flatten_5 (Flatten)         (None, 2048)              0

 dense_26 (Dense)            (None, 512)               1049088

 dense_27 (Dense)            (None, 10)                5130

=================================================================
Total params: 24641930 (94.00 MB)
Trainable params: 1054218 (4.02 MB)
Non-trainable params: 23587712 (89.98 MB)
_________________________________________________________________
```

Figure 4: Results Table-DvC is Dogs vs. Cats and HvH indicates Horses vs. Humans

Model	Train Accuracy	Train Loss	Validation Accuracy	Validation Loss	Test Accuracy	Test Loss	Process Time (s)
LeNet5-Cifar	0.594133	1.151153	0.536800	1.318594	0.525100	1.352243	3015
VGG16-Cifar	0.729600	0.767658	0.613600	1.125642	0.616300	1.154595	3179
InceptionV3-Cifar	0.800578	0.560641	0.632200	1.274827	0.633400	1.272536	2559
ResNet50-Cifar	0.384143	1.714115	0.379012	1.709711	0.383900	1.704746	3015
LeNet5-DvC	0.485625	0.695398	0.495000	0.693617	0.493600	0.693677	1134
VGG16-DvC	0.898750	0.226135	0.916875	0.198148	0.929400	0.173969	9526
InceptionV3-DvC	0.976562	0.057199	0.976250	0.059125	0.980800	0.045138	1821
ResNet50-DvC	0.635364	0.692435	0.673125	0.771967	0.670800	0.761165	4707
LeNet5-Eurosat	0.538323	1.304605	0.462037	1.512242	0.480185	1.457089	298
Vgg16-Eurosat	0.937500	0.182745	0.864815	0.467281	0.877407	0.404448	4865
InceptionV3-Eurosat	0.930041	0.201293	0.809722	0.760171	0.823148	0.688543	1871
ResNet50-Eurosat	0.573765	1.173917	0.580093	1.146246	0.580000	1.117187	2406
LeNet5-HvH	0.532847	0.691226	0.392157	0.709780	0.475728	0.698618	55
VGG16-HvH	1.000000	0.000351	0.990196	0.036456	0.990291	0.039230	768
InceptionV3-HvH	0.997567	0.014266	0.990196	0.029381	1.000000	0.000036	175
ResNet50-HvH	0.964720	0.146163	0.911765	0.195347	0.932039	0.183179	277