Classifying NFL Team Logos

Dan Zylkowski

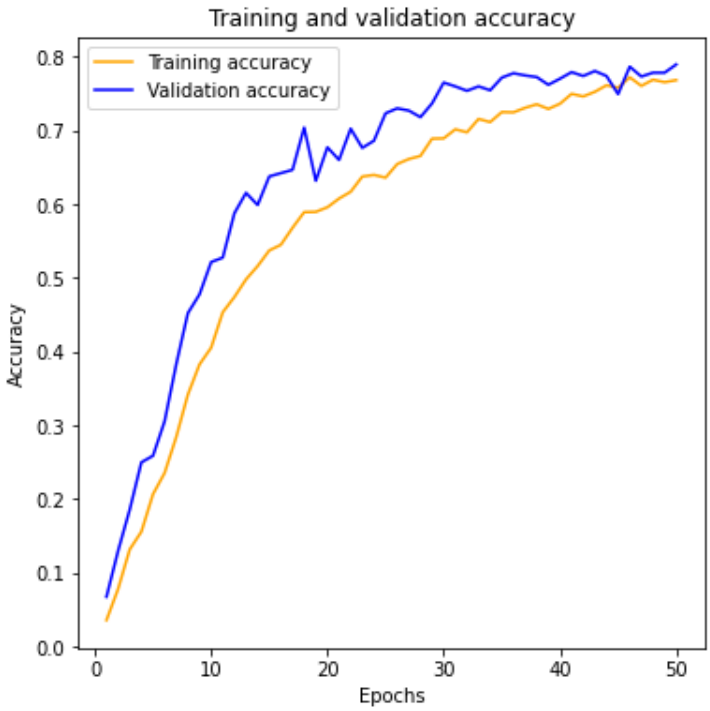
DSC 680

Fall 2021

Abstract/Executive Summary

The NFL is the most popular sport in the United States. For my project I decided to build a model to classify the logos of NFL teams. Since traditional computer programming is not a viable solution for image classification, I decided to use deep learning to solve the problem. To accomplish the image classification task, a deep learning convolutional neural network (CNN) was built from scratch and trained 5,463 training images and 1,350 validation images that were acquired from the Bing Images search engine. The images were distributed approximately evenly across the 31 classes to ensure that each team was similarly represented in the dataset. Accuracy was the metric that was chosen to optimize, as we are trying to optimize the correct classification of the images. After 50 training epochs, the validation accuracy reached 78.9%, which was a decent score for a CNN model built from scratch using a dataset containing just over 200 images in each class. This accuracy score was used as a baseline for the project.

Figure 1: Training and validation accuracy of a baseline CNN model



To improve on the baseline score and overcome the limited size of the dataset, an additional model was built on top of the pre-trained ResNet50[1] model architecture using the fastai[7] python library. The ResNet50 model contains 50 total layers and has been trained on millions of images with a high degree of accuracy. Since deep-learning models are highly repurposable we can take the ResNet50 model that was trained on a large-scale dataset and reuse it on our dataset with only a few changes. A 91% validation accuracy was achieved by using the ResNet50 model without any tuning of the model. When we tuned the ResNet50 model, the validation accuracy increased to 96.8%. Nest, the fastai library helped to identify and remove noisy images and one incorrect image. Finally, we used the updated data to build a model with an excellent validation accuracy of 98.8%. This is a nearly 20% improvement over the baseline model accuracy of 78.9%.

Figure 2: Final validation accuracy results using the ResNet50 model after tuning and data cleaning

epoch	train_loss	valid_loss	accuracy	time
0	0.250597	0.052429	0.986188	01:41
1	0.288559	0.034385	0.988029	01:41
2	0.196876	0.035487	0.988029	01:44

Problem Statement

Image classification is an exciting subclass of computer vision. Important uses cases for image classification include identifying cancer in X-rays or allowing self-driving cars to identify objects in the road. For this project the use case could be to enable online retailers to identify the name of the team based on a picture of a logo. This project aims to classify the images of the logos of 31 of the 32 NFL teams. The Washington Football Team is not being included in this analysis since the team has no current logo and will not have a new logo until 2022. This project will use deep learning to create a robust CNN model that will accurately classify NFL team logo images.

Methods

Data Collection

To collect the NFL team images, the bing_image_downloader[12] library. This library allows you to specify a search query and number of results, and it automatically downloads the images to the specified directory. While downloading the images was easy, not all images were correctly classified by the search engine. As a result, I used multiple related search terms and collected the data in one folder. Then, I reviewed each image individually to remove duplicate

same-sized images and images that were incorrectly classified. After carefully curating the image dataset, 5,463 training images and 1,350 validation images were uploaded to google drive for use in google colab. The data collection steps are contained in a jupyter notebook separate from the modeling notebook. The data collection steps can be easily modified for other image classification projects. The link to the final training images can be found on google drive.

Figure 3: NFL team logos

Building a baseline CNN model

Using the Keras library, A CNN model was built using six convolutional and max-pooling layers, as well as two dense and activation layers on top. Since this is a multi-class classification problem, the categorical cross-entropy loss function was used when compiling the model and the softmax function was used for the final activation function.

When using neural networks for image classification, a large amount of data is often required. This is because neural networks can contain thousands or millions of parameters that

need to be trained. To help with the lack of data, I used image augmentation to synthetically add additional data to the current dataset. Common augmentation techniques include flipping, rotating, scaling, cropping, and translating current images to create new images. The neural network will consider the augmented images as being new data, and this will help train the network to learn from different parts of the image.

Two dropout layers were also added to the model to avoid overfitting. The idea behind using dropout layers is that in each learning step we select a random subset of the neurons in a layer and ignore it on both the forward pass and in the back-propagation of error. By randomly turning off neurons in the layer, we prevent the model from learning random noise in the data, and thus helps to avoid overfitting. The dropout layers were the reason the validation accuracy was higher than the training accuracy across nearly all epochs. The best validation accuracy achieved using the baseline model was 78.9%.

Building a model on top of the ResNet50 model architecture

Using the fastai library, I froze the convolutional layers of the ResNet50 model and added dense layers on top of it. The idea behind this strategy is to keep all the generalized skills of the ResNet50 model, and to add train a few dense layers on top of it using the dataset. The learning rate was set to a very low value since we are only trying to fine-tune the model performance. After building a model on top of the ResNet50 architecture, I was able to use fastai to identify images with the largest loss values. These images were displayed, and I was then able to either remove or reclassify them to improve model performance. The result was that the validation accuracy increased nearly 8%, from 90.9% to 98.8%. These results are a vast

improvement over the baseline score and show that the model is skillful at classifying NFL team logos.

Limitations

While the model had a high validation accuracy of 98.8%, the main limitation of such a result is the limited number of images that were used to train the model. While this is an obvious limitation, I would counter that there are not that many representations of a 2-dimensional NFL team logo, and most of the logos are different by design. Another potential limitation is that as team logos change, the model would need to be trained on the new logos.

Discussion/Conclusion

To recap, we first built a CNN from scratch in Keras and achieved a validation accuracy of 78.9%. Then, we used the fastai library to build a baseline model on top of the ResNet50 model and achieved a validation accuracy of 91%. We then tuned the model and the validation accuracy increased to 96.8%. Next, we used the fastai toplosses function to remove noisy images and one incorrect image. Finally, we used the updated data to build a model with an excellent validation accuracy of 98.8%.

Overall, I am happy with the increases in accuracy for the different models. While I have some previous experience in Keras, this project was my first time using the fastai library, and I was impressed with the results. Fastai is a powerful low code library that uses PyTorch to easily build deep learning models.

The most challenging part of the project was curating the image datasets. While I found a programmatic solution for identifying and deleting images of the same size, I could not find a solution for identifying images that were incorrectly labeled by the image search engine. As a result, it was necessary to manually view each image to filter out the incorrectly labeled images, otherwise model accuracy would be limited by the image search engine's model accuracy.

Next Steps

Due to the general format of this project, it could be adapted and used for other image classification projects. As long as the class labels are properly defined and the images are saved in the train folder during the data collection phase, the class label names can be inferred automatically. Thus, the modeling notebook can be used for a new image classification project with minor edits to the CNN and fastai model architectures.

References:

1. Boesch, G. (2021, August 29). *Deep residual networks (ResNet, RESNET50) - guide in 2021*. viso.ai. Retrieved October 3, 2021, from https://viso.ai/deep-learning/resnet-residual-neural-network/.
2. Brownlee, J. (2019). *Better Deep Learning.*
3. Brownlee, J. (2019, August 19). *Crash course in Convolutional Neural Networks for machine learning*. Machine Learning Mastery. Retrieved October 3, 2021, from https://machinelearningmastery.com/crash-course-convolutional-neural-networks/.
4. *2021 NFL logos - National Football League logos - the news and history of sports logos and uniforms*. 2021 NFL Logos - National Football League Logos - Chris Creamer's Sports Logos Page - SportsLogos.Net. (n.d.). Retrieved October 3, 2021, from https://www.sportslogos.net/teams/list_by_year/72021/2021_NFL_Logos/.
5. Chollet François. (2018). *Deep learning with python*. Manning Publications Co.
6. Chollet, F. (n.d.). *The keras blog*. The Keras Blog ATOM. Retrieved October 3, 2021, from https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html.
7. Fast Ai. · Making neural nets uncool again. (n.d.). Retrieved October 3, 2021, from https://www.fast.ai/.
8. Gaydos, R. (2020, July 13). *Washington Redskins retire team name, logo; no replacement announced*. Fox News. Retrieved October 3, 2021, from https://www.foxnews.com/sports/washington-redskins-retire-team-name.

9. Kaushik, A. (2020, July 21). *Understanding Resnet50 architecture*. OpenGenus IQ: Computing Expertise & Legacy. Retrieved October 18, 2021, from https://iq.opengenus.org/resnet50-architecture/.
10. Osinga, D. (2018). *Deep learning cookbook practical recipes to get started quickly*. O'Reilly.
11. Washington football team teases new name and rebrand for 2022. (n.d.). Retrieved October 3, 2021, from https://www.msn.com/en-us/sports/nfl/washington-football-team-teases-new-name-and-rebrand-for-2022/ar-BB1dWTzS.
12. *Bing-image-downloader*. PyPI. (n.d.). Retrieved October 18, 2021, from https://pypi.org/project/bing-image-downloader/.

Appendix

Figure 4 : Confusion Matrix for the final model