**Training and Analyzing Intelligent Systems on CIFAR-100**

The purpose of this project was to design, train, and analyze three best performing neural network types on the CIFAR-100 image classification dataset. Those network types included building convolutional, artificial, and random forests using the tools available with TensorFlow, TFLearn, and SKLearn. This report shows the findings, successes and failures, the program process, and reflections from building the project and training those networks. For more information and details on the project itself, please refer to the README.md file included with the project supplied with this report or visit the hosted repository <https://github.com/itsjaboyd/cifar100-project> through GitHub.

**CIFAR-100 Dataset**

The CIFAR-10 dataset can be found and downloaded at <https://www.cs.toronto.edu/~kriz/cifar.html>. Quickly overviewing the dataset, CIFAR-100 consists of 100 classes containing 600 images each (500 training images and 100 testing images) for a total of 60,000 images. Those 100 classes are divided into 20 super-classes. Each image is labeled with its immediate and super classes. CIFAR-100 exists as a subset of the 80 Million Tiny Images dataset that serves as a major dataset to perform object image classification techniques on. The dataset has somewhat of a twin known as CIFAR-10 in a much tighter knit and closer identification realm, including just 10 classes with 6,000 images each instead [3]. Some example images are given by the dataset below.





CIFAR-100 has been a rather popular dataset to work with in devising highly accurate convolutional neural networks. For example, the latest and greatest model for this dataset comes from Alexander Kolesnikov et al. using a newly proposed method known as Big Transfer (BiT) and other heuristics for achieving their highly accurate network [1]. The second highest accuracy has been achieved by Mingxing Tan and Quoc V. Le proposing a new scaling method and highly effective compound coefficient on already existing networks [2]. There exist many other impressive and notable network architectures on the CIFAR-100 dataset that are interesting to review, but of course being extremely popular in the image classification community, there are a plethora of classification methods out there.

**Convolutional Neural Network**

My findings with CNN

**Artificial Neural Network**

My findings with ANN

**Random Forest** **Network**

Everything I tried/why I couldn’t make random forests work

I hate random

Stupi ass

Random forest bigtches

**Project Reflection**

More than a couple things were learned throughout this entire project experience. First and foremost, attempting to accurately estimate the time needed to build the foundational features of the project should have been given priority. Second, sometimes TensorFlow was rather unhelpful in outputting error messages and helping debug buggy code. Finally, rigorously sticking to the project schedule should have been one of the top priorities as well, given the circumstances and breaks in school.

Before beginning development on the project and as outlined in the project proposal, it seemed as though the structure and flow of the project was going to be simple. Three files for three different network types were going to be enough to organize the project, but after realizing that some code could be reused by other networks and unit testing could be helpful and how the project was going to be run exactly, re-structuring the project was necessary. A reiteration of the requirements phase was necessary to effectively plan out what was needed to complete, which minorly put back the project schedule. Allocating more than enough estimated time to gather requirements and seriously plan out exactly what the project should and should not perform is vitally important. It was extremely easy to get sidetracked and caught up in making cool but irrelevant features instead of focusing on the main task ahead.

While TensorFlow has decent documentation about using the framework, there were walls to push through and external research to be done. The biggest friend of a developer who is training and loading multiple different networks is the “reset\_default\_graph()” function. After what felt like days of trouble shooting checkpoint errors and fine-tuning network architectures, finally coming upon and using this function was huge in getting development back on track. At one point it was thought that completely retraining the saved convolutional network was necessary, but it in fact was not, which wasted time. Along with knowing the tools being used, doing adequate research into what other tools that could be used would have been helpful as well. TensorFlow ships with an API known as Keras for deep learning purposes that definitely could have come in handy in network analysis and training.

Every software developer should plan for setbacks within the project schedule, but between planning out milestones and deliverables, sticking to the schedule should be the number one goal. Over the project timeline was Thanksgiving break which was approximately a week off from school. Knowing myself and the nature of taking time of from school, absolutely no work was done over this Thanksgiving period, which of course pushed back the project timeline by another week. Luckily, there was around another two weeks budgeted towards the end of the timeline to catch up and work on the required documentation, but that week wasted was valuable development time.

While not every aspect of the project was completed, the project was extremely interesting and rewarding seeing these different types of networks perform in image classification. Even though the due date has come, project development will most likely not come to an end. Seeing the results of training a random forest on CIFAR-100 and comparing its results to the convolutional and artificial networks will be insightful and rewarding.

**Conclusion**

Final words on the application of network types on cifar100

**Works Cited**

[1] Kolesnikov, Alexander et al. “Big Transfer (BiT): General Visual Representation Learning.” ECCV (2020). https://arxiv.org/abs/1912.11370v3.

[2] Tan, M. and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.” ArXiv abs/1905.11946 (2019): n. pag.

[3] Krizhevsky, A. (n.d.). CIFAR-10 and CIFAR-100 Datasets. Retrieved from https://www.cs.toronto.edu/~kriz/cifar.html