**Final Project**

- choose a dataset from a Kaggle.com competition.

- make a research data analysis project using multiple learning approaches

- use your own initiative to define one or several research questions

- use the resources available to apply different approaches, using any Python libraries, to this data to predict accurately the class or the value of the unlabeled data

- to answer your research questions

Pages: 8 - 10 pages, similar to a research paper for a publication

- this implies a state of art techniques

- a presentation of the research questions

- the chosen methods to tackle them

- a presentation of the results, discussion, conclusion/future work

- attach a description of the participation of each student to the project

**Note:** You do not have to develop a completely new approach, but you have to make experimentations/comparisons and present your results as it is a new method conceives by you.

"Epoch X/Y": This indicates the current training epoch out of the total number of epochs (Y). An epoch is one complete pass through the entire training dataset.

"1563/1563": These numbers represent the number of batches processed in the current epoch. In this case, you have 1563 batches, and it means that each epoch consists of 1563 iterations (or mini-batches) through the training data.

"- 8s 5ms/step": This part provides information about the time it took to complete the current epoch. "8s" indicates that this epoch took 8 seconds to complete, and "5ms/step" indicates that each batch (step) took an average of 5 milliseconds to process.

"loss: 1.5686 - accuracy: 0.4381": These are the training metrics for the current epoch. "loss" is the value of the loss function, which measures how well the model is performing (lower is better). "accuracy" is the classification accuracy on the training data for this epoch.

"val\_loss: 1.2423 - val\_accuracy: 0.5595": These are the validation metrics for the current epoch. "val\_loss" is the value of the loss function on a separate validation dataset (not used for training), and "val\_accuracy" is the classification accuracy on the validation data for this epoch. These metrics are used to assess how well the model generalizes to unseen data.

The log represents the progress of training a neural network over multiple epochs, showing the loss and accuracy on both the training and validation datasets. The goal is to reduce the loss and increase the accuracy on the validation set, indicating that the model is learning to make better predictions.

model.compile(loss='sparse\_categorical\_crossentropy',

optimizer=tf.keras.optimizers.Adam(learning\_rate=0.01), metrics='accuracy')

history = model.fit(x\_train, y\_train, epochs=25, validation\_data=testing\_generator, batch\_size=128, shuffle=True)

A screen shot of a computer screen

Description automatically generated

A logo of a university of windsor

Description automatically generated

Exploring the Effectiveness of Deep Learning Architectures in Multiclass Classification Task Using the CIFAR-100 Dataset

***Akshat Sharma Justin Neal George Kaceli***

[*insert@uwindsor.ca*](mailto:insert@uwindsor.ca)[*insert@uwindsor.ca*](mailto:insert@uwindsor.ca)[*kaceli@uwindsor.ca*](mailto:kaceli@uwindsor.ca)

December 14th, 2023

**Abstract**

**Introduction**

The CIFAR-100 Image Classification Project introduces the goal of building and training Convolutional Neural Networks (CNNs) for classifying images within the CIFAR-100 dataset. Each image is assigned a label, and these labels are integer values representing one of the 100 classes. This presents a multi-label classification challenge. Hyperparameter tuning is used to analyze the strengths and weaknesses of the models. The objective of this research is to explore how advanced machine learning techniques can enhance the accuracy of image classification in the CIFAR-100 dataset.

**Dataset**

This dataset is comprised of 60,000 32x32 color images distributed across 100 diverse classes. It contains many images across 100 non-overlapping classes. Since the dataset contains 60,000 samples in total, this results in each class only has 600 samples.

|  |  |
| --- | --- |
| **Superclass** | **Classes** |
| Aquatic mammals | Beaver, dolphin, otter, seal, whale |
| Fish | Aquarium fish, flatfish, ray, shark, trout |
| Flowers | Orchids, poppies, roses, sunflowers, tulips |
| Food containers | Bottles, bowls, cans, cups, plates |
| Fruit and vegetables | Apples, mushrooms, oranges, pears, sweet peppers |
| Household electrical devices | Clock, computer keyboard, lamp, telephone, television |
| Household furniture | Bed, chair, couch, table, wardrobe |
| Insects | Bee, beetle, butterfly, caterpillar, cockroach |
| Large carnivores | Bear, leopard, lion, tiger, wolf |
| Large man-made outdoor things | Bridge, castle, house, road, skyscraper |
| Large natural outdoor scenes | Cloud, forest, mountain, plain, sea |
| Large omnivores and herbivores | Camel, cattle, chimpanzee, elephant, kangaroo |
| Medium-sized mammals | Fox, porcupine, possum, raccoon, skunk |
| Non-insect invertebrates | Crab, lobster, snail, spider, worm |
| People | Baby, boy, girl, man, woman |
| Reptiles | Crocodile, dinosaur, lizard, snake, turtle |
| Small mammals | Hamster, mouse, rabbit, shrew, squirrel |
| Trees | Maple, oak, palm, pine, willow |
| Vehicles 1 | Bicycle, bus, motorcycle, pickup truck, train |
| Vehicles 2 | Lawnmower, rocket, streetcar, tank, tractor |

A collage of images of animals

Description automatically generated

In the context of the CIFAR-100 dataset, each image is represented as a three-dimensional array with dimensions 32 x 32 x 3. There is no concept of columns in the traditional tabular sense. Instead in the context of image data, the number of features corresponds to the dimensionality of the input space. This means that the dimension is determined by the size and colour channels of the images. Where 32 x 32 is the spatial resolution which is composed of the height and width. While the 3 corresponds to the three-colour channels representing red, green, and blue.

**Models**

The neural network architecture is encapsulated within the Network class. This is achieved through a dynamic import mechanism, where the script imports the desired model architecture from a module by the model’s variable name. The flexibility of dynamically importing the network class allows for a modular and configurable approach to building different neural network architectures based on the specified model names. The model variable holds the constructed neural network model, and subsequent operations, such as displaying the architecture summary, compiling the model, and training, are performed using this instantiated model.

**Data Preprocessing**

Upon loading the data, the initial action is to preprocess it. This is handled by a preprocessing function that scales the pixel values to a range between 0 and 1 by dividing each value by 255 and returning the results. Once the model is successfully imported, an instance of the network is created, and the model is built. Post-training, important metrics such as training and testing loss, along with accuracy values, are used from the training history for subsequent analysis.

**Models**

Dense

CNN

RESNET

VGG

Model Training and Evaluation

Conclusion

Future Work

-pca model

Description of Student Participation

References

<https://github.com/christianversloot/machine-learning-articles/blob/main/how-to-build-a-convnet-for-cifar-10-and-cifar-100-classification-with-keras.md>