**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.

The chosen dataset consists of 60,000 color images, each measuring 32 x 32 pixels and having 3 color channels. These images are categorized into 10 different classes, with each class containing 6,000 images. The dataset is divided into a training set with 50,000 images and a test set with 10,000 images. It is a multi-label classification problem.

"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, where the objective is to develop a model capable of learning intricate patterns and features within the images and the model is expected to predict the appropriate class.

**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 color 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-color channels representing red, green, and blue.

**Models**

**Data Preprocessing**

The class labels (y\_train and y\_test) are then one-hot encoded using tf.keras.utils.to\_categorical to facilitate training with the categorical crossentropy loss. The pixel values of the images (x\_train and x\_test) are normalized to the range [0, 1] by dividing them by 255.

-dense

-cnn model

-resnet

The CIFAR-100 dataset, being very complex as discussed earlier causes a lot of challenges for basic models, along with the small size of the images cause significant problems for basic models like a vanilla CNN to extract meaningful data for classification. To combat the issue, we try to harness the power of pre-trained models via Transfer Learning, which is the process of utilizing a model developed for a one task to be reused as a starting point for a model on a second task. It is useful in scenarios where there is a limited amount of training data or other limitations such as hardware and so on. We chose to use ResNet50 (residual networks) which is a popular architecture known for its ‘Residual blocks’, which help in training very deep networks by addressing the vanishing gradient problem, which is one of the key challenges in training very deep neural networks, where gradients become too small to make any significant change in the weights during backpropagation, hindering the learning process. ResNet addresses this through its innovative residual learning framework. By introducing shortcut connections (also known as skip connections) that bypass one or more layers, ResNet allows the training of much deeper networks. These connections enable the network to learn an identity function, ensuring that higher layers can perform at least as well as lower layers. This is crucial for learning complex patterns in data-rich environments like CIFAR-100. The dataset also comprises of 100 different classes, giving it a very high categorical complexity. The depth of ResNet allows it to learn a rich hierarchy of features. In image classification tasks, lower layers often learn basic features like edges and textures, while deeper layers learn more complex features specific to the objects in the dataset. This hierarchical learning is particularly important for CIFAR-100, where the ability to discern fine-grained differences between similar categories can significantly impact classification accuracy. Despite its depth, ResNet is relatively efficient to train. The residual connections help in propagating gradients throughout the network, facilitating faster and more effective training. This efficiency is crucial when working with extensive datasets like CIFAR-100, where training time and computational resources can be significant constraints.

To modify the ResNet50 architecture for our use case we froze the layers and initialized it with ImageNet weights and the input shape of 224,224,3 which it was trained on. We also made it so that the batch normalization layers were trained, this is because batch normalization helps as we go deeper into the network and helps the model converge faster.

The **ResNet50** model was trained on ImageNet, where images are **typically 224x224 pixels**. In contrast, the CIFAR-100 dataset consists of 32x32 pixel images. Directly inputting these smaller images into **ResNet50** would be inappropriate due to the size mismatch. The UpSampling2D layer that we added, addresses this by resizing the smaller images to a size compatible with ResNet50, and through that we feed our inputs into the ResNet model, with the outputs from it being fed to a **GlobalAveragePooling** layer, which reduces the spatial dimensions of the feature maps to a single value per feature map. We finish the model by adding dropout layer and a **Dense 256 layer** with **relu** and finish it off with batch normalization and output it using a **Dense 100 layer** with **softmax** to since we are multiclass classification.

-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>