**Final Project ADTA 5550 Deep Learning with Big Data**

Student Name: Yog Chaudhary

Professor: Dr. Hamidreza Moradi

**PART I: Use TensorFlow Directly in Coding (5 Points)**

**Question 1.1: Is the student required to use TensorFlow directly in coding (build, train, and test CNN) in this homework assignment**?

It is mentioned by the Professor to use TensorFlow directly in the Assignment as after mid-term we would be using this instead of Keras. There are advantages of using Tensor flow:

* Customization and Low-Level Control
* Research and Experimentation: TensorFlow provides the flexibility to implement and test new ideas quickly.
* Production Deployment: TensorFlow Serving, TensorFlow Lite, and TensorFlow.js are tools and libraries designed for deploying models on servers, mobile devices, and web applications. If you are building production-ready applications, TensorFlow may be the most suitable choice.
* Performance Optimization:
* Community and Ecosystem: TensorFlow has a large and active community.
* Integration with TensorFlow Extended (TFX): TensorFlow Extended (TFX) provides a seamless solution.
* Compatibility and Futureproofing: TensorFlow is the core framework developed and maintained by Google.

**Question 1.2: Should the student use Keras in coding (build, train, and test CNN) in this homework assignment?**

Keras is a tool for creating deep learning models, such as CNNs, with minimal code. It’s built on frameworks like TensorFlow, It’s the process of setting up, training, and testing neural networks. Ideal for beginners, Keras offers pre-built models and layers for easy CNN architecture creation. It runs on both CPU and GPU, offering hardware flexibility. However, Keras may not provide access to all TensorFlow's advanced features and abstract some details, which could limit control over models.

Keras several disadvantages of TensorFlow are following.

* Limited Low-Level Control
* Research Flexibility
* Limited Deployment Options
* Performance Optimization: For optimizing performance on specialized hardware like GPUs and TPUs, TensorFlow's low-level control can be advantageous.
* Community and Ecosystem: Keras, while well-supported within TensorFlow, might have fewer specialized resources.
* Compatibility and Future Updates: While Keras is integrated into TensorFlow, using TensorFlow directly ensures compatibility with the latest advancements and updates in the TensorFlow ecosystem.
* Complex Pipelines

**PART II: A Dataset of Images or Audio Files (10 Points)**

Dataset Name: Stanford Dogs Dataset

<http://vision.stanford.edu/aditya86/ImageNetDogs/>

Dataset Report: Stanford Dogs Dataset

Official Website: Stanford Dogs Dataset

Download Link: The dataset can be downloaded from the official website by following the provided links to the "All images" and "Annotation" sections. The data is available in compressed format.

**Data Details:**

**Type:** Image dataset.

**Labeled:** Yes, this dataset contains labeled images of dogs belonging to various breeds.

**Usage for Deep Learning**: The dataset is suitable for deep learning research, particularly in image classification, object detection, and fine-grained recognition tasks.

**Dataset Overview:**

The Stanford Dogs Dataset is a collection of images of dogs belonging to *120 different breeds*. It is intended for various computer vision tasks, such as image classification, fine-grained recognition, and object detection. This dataset is a subset of the larger ImageNet dataset, focusing exclusively on dog breeds.

**Data Structure:**

The dataset is structured as follows:

Image Data: The primary component of the dataset consists of images of dogs. These images are organized into subdirectories based on their respective breed labels. Each image is labeled with the breed name, making it suitable for supervised learning tasks.

Annotations: Along with the images, the dataset provides annotation files in XML format. These annotation files contain detailed information about the bounding boxes of dog instances within the images, which can be useful for object detection and localization tasks.

**Number of Items:**

The Stanford Dogs Dataset contains a total of 20,580 images of dogs belonging to 120 different breeds. Each breed category is represented by various images, ensuring diversity within the dataset.

**Usage and License:**

The dataset is intended for research purposes and is available for use without restrictions. However, it is essential to review the licensing and usage terms provided on the official website to ensure compliance with any specific requirements.

**Conclusion:**

The Stanford Dogs Dataset is a valuable resource for researchers and practitioners in the field of computer vision and deep learning. With a diverse set of images covering 120 different dog breeds, it offers ample opportunities for various image analysis tasks. Researchers can access this dataset through the official website, download the data, and explore its potential for their specific research projects.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**References:**

Citation: <http://vision.stanford.edu/aditya86/ImageNetDogs/>

GitHub Link: <https://github.com/ayushdabra/stanford-dogs-dataset-classification>

**PART III: Obtain CIFAR-10 Dataset (5 Points)**

First, download the CIFAR dataset from the canvas module. Then, Start a Remote Virtual Server. If the remote instance does not run, Start the remote virtual server.

* Screenshot Hare

A screenshot of a computer

Description automatically generated

Start an **SSH session**. Then I Click open in a browser window.

* First, open the SSH remote server, displaying a new window. Follow the instructions shown in the screenshot provided. Once you have set up CIFAR\_10\_DATA, upload the datasets.
* Go to the JP\_NTBK folder and run the command line "cd JP\_NTBK" to access it.
* Create a new sub-folder named "DATA" within the JP\_NTBK folder. To do this, run the command line "mkdir DATA".
* To verify that the new sub-folder has been successfully created, run the command line "ls -l".
* Copy the data file into the newly created sub-folder by running the command line to copy 7 files for CIFAR\_10\_DATA

A screenshot of a computer screen

Description automatically generated

CIFAR dataset was effectively uploaded to the designated folder. Consequently, both the downloading of the dataset from the canvas module and the subsequent uploading of the dataset to the remote server were accomplished.

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

* Successfully Jupyter Notebook Run and uploaded to a remote server.

**PART IV: Build, Train, and Test CNN on CIFAR-10 Dataset (30 Points)**

**Design the convolution neural network used for the project.**

**Design:** In this project, the chosen image classification technique is CNN, a method that applies filters to the raw pixel data of an image to extract features that are not achievable through traditional neural networks.

* The CNN architecture comprises two convolutional layers, a pooling layer, and fully connected layers, reducing complexity and computational load while enhancing image recognition capabilities.
* Despite its strengths, CNN has limitations such as the need for large datasets, a lack of spatial invariance to input data, and the intricacies of hyperparameter tuning.
* The CNN model utilized in this project employs multiple convolutional operations with varying filter sizes, incorporates max pooling, applies the ReLU activation function, and employs batch normalization.
* Following the convolutional operations, the data is flattened, followed by several fully connected layers with different neuron counts, culminating in a fully connected layer comprising 10 units, corresponding to the number of distinct images.

A screenshot of a computer

Description automatically generated

Diagram

Description automatically generated

**A white box with black text

Description automatically generated**

**Dataset with the CIFAR\_10\_Data**

* CIFAR-10 is comprised of 60,000 color images, each measuring 32x32 pixels. These images are distributed across 10 distinct categories, with each category containing 6,000 images. For training, 50,000 images are allocated, leaving the remaining 10,000 images for testing.
* The dataset is structured into 5 training batches and 1 test batch, each holding 10,000 images. The test batch is thoughtfully composed of 1,000 images randomly selected from each class. The images in the training batches are arranged in a random order, and certain batches might have a varied number of images from one class in comparison to another. Nonetheless, when all the training batches are combined, there are 5,000 images from each class.
* In total, the dataset encompasses 10 distinctive classes, and as an illustrative example, a selection of 10 random images from each class is provided.

A collage of images of animals

Description automatically generated

The dataset contains separate classes, and there is no overlap between the categories of automobiles and trucks. The "automobile" class encompasses various types of vehicles such as sedans, SUVs, and similar models, whereas the "truck" class exclusively comprises large trucks, excluding pickup trucks.

References: CIFRA, Link <https://www.cs.toronto.edu/~kriz/cifar.html>

A screenshot of a computer program

Description automatically generated

**Displaying Individual Sample Images:**

* To present the data visually, we employed the "matplotlib" library.
* Additionally, we imported the "numpy" library, which is versatile and operates with zero-based indexing like sequences.
* As a part of preparing image data for input into a CNN model, reshaping is undertaken. This process necessitates the input tensor dimensions to be either (width \* height \* num\_channel) or (num\_channel \* width \* height).
* ransposition is carried out to rearrange the tensor dimensions as required.

A screenshot of a computer

Description automatically generated

A blurry image of a dog

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Rearranging Data Using Functions:**

* to encode CIFAR-10 dataset image labels, one hot encoding is employed.
* A vector of size 10 is essential to accurately represent the 10 classes of CIFAR-10 images.
* The following code snippet, along with a corresponding illustration, is supplied to elucidate this procedure.

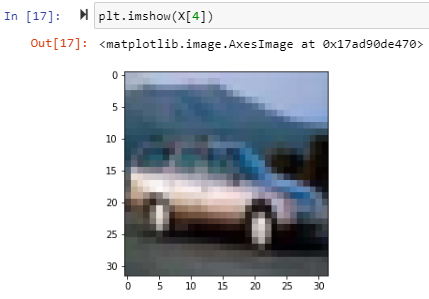
A screenshot of a computer program

Description automatically generated

Table

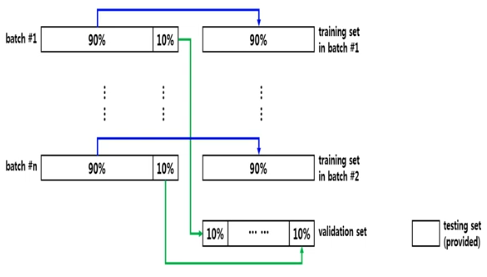
Description automatically generatedTable

Description automatically generated

* Here it clearly illustrates the original label data and the one hot encoded format.

**Now, we move on to the data preprocessing phase.**

* To form the validation dataset, 10% of the data from each cluster will be pooled together, leaving the remaining 90% of the data designated for the training dataset.



**Constructing the Model: Key Steps:**

The model-building process involves three significant stages:

* In a convolutional neural network, a filter serves the purpose of preserving a single pattern.
* The stride parameter is instrumental in dictating the extent by which the filter window advances with each step during the convolution process.
* To maintain an output length identical to the input post-convolution, the padding parameter is configured as 'same.' The structure takes the form of (height \* width \* number of filters).

A screenshot of a computer program

Description automatically generated

**Creating a Convolutional Neural Network (CNN)**

* The ReLU activation function is applied to transform an input value into a new value within the range of 0 to positive infinity.
* When the input value surpasses a certain threshold, the output increases linearly.
* The output swiftly reaches its maximum value, which is 0, as the input value exceeds the threshold.
* For image downsizing by a specified scale, max pooling is implemented.
* To prevent overfitting, the incorporation of dropout is a viable approach.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer code

Description automatically generated

**Establishing the First and Second Convolutional Layers**

* The output from the initial convolutional layer serves as the input for the subsequent convolutional layer. Moreover, a second pooling layer is introduced to diminish the spatial dimensions.

A screenshot of a computer

Description automatically generated

**Reshaping and flattening the data.**

A screenshot of a computer

Description automatically generated

Once the data is reshaped and flattened, we introduce a dropout layer to mitigate the risk of overfitting in the image data.

* Subsequently, we proceed to define the loss function, which is followed by the computation of the softmax activation.
* For datasets like CIFAR-10, where the objective is to evaluate loss across 10 distinct classes, the softmax activation function is employed.

**Optimizers:**

* During the training of the neural network, the primary objective is to minimize the cost using a designated algorithm. To accomplish this, optimizers such as Adagrad and Adam can be effectively utilized.

A screenshot of a test

Description automatically generated

**Training and Testing:**

* The "train\_neural\_network" function is responsible for conducting optimization operations on a data batch as part of the training procedure. This function is intended to be employed iteratively across various epochs and batches.
* The execution of these tasks is managed through the "Session.Run" method.

A screenshot of a computer program

Description automatically generated

**Results**

A white background with black text

Description automatically generated A screenshot of a computer

Description automatically generated A white background with black text

Description automatically generated A white background with black text

Description automatically generated A white background with black text

Description automatically generated

**PART V: Compare Convolutional Neural Network Performance (10 Points)**

**A 6-layer CNN for MNIST has the following architecture:**

* Input Layer: This is the initial layer where the 28x28-pixel grayscale MNIST images are provided as input.
* Convolutional Layer 1: The first convolutional layer typically consists of 32 or more filters, each with a small receptive field (e.g., 3x3), and applies the rectified linear unit (ReLU) activation function.
* Max-Pooling Layer 1: Following the first convolutional layer, a max-pooling layer is applied to downsample the feature maps.
* Convolutional Layer 2: The second convolutional layer often has more filters (e.g., 64) and applies ReLU activation.
* Max-Pooling Layer 2: Like the first max-pooling layer, this layer reduces the spatial dimensions of the feature maps.
* Fully Connected Layers: After the convolutional and max-pooling layers, the feature maps are flattened and connected to one or more fully connected layers. A common setup is a 64-neuron hidden layer followed by a 10-neuron output layer with a softmax activation function for classifying the 10 digits (0-9).

|  |  |
| --- | --- |
| MNIST | **CIFAR** |
| Accuracy is 90% above after 200 | Accuracy doesn’t reach above 70% |
| The final Accuracy is 98% | Final Accuracy is 65% |
| The test accuracy is 98% | The test accuracy is 10% |
| Simple CMM with 2 conv layers is fitting data to 98% | The model with 14 layers is also not good for making an accuracy of 90% |

CIFAR-10, renowned for its variability in the spatial positioning and orientation of objects, can be prone to overfitting when intricate models are employed. Moreover, the added complexity of color images introduces an extra layer of challenge compared to grayscale images.

* In stark contrast to the MNIST dataset, which solely encompasses single-dimensional handwritten digits, CIFAR-10 portrays 3D objects that can be observed from diverse angles. This implies that a more extensive training dataset is essential to attain effective generalization.
* The presence of background elements within CIFAR-10 images can occasionally lead to misidentification, resulting in object recognition confusion.
* Unlike the MNIST dataset, where the classes maintain a fixed shape, CIFAR-10 objects within the same category can exhibit variations in features, rendering the generalization process notably more demanding.
* Training time efficiency exhibits a trend analogous to testing accuracy in both datasets. Higher batch sizes translate to extended training durations, but they also correlate with elevated testing accuracy.
* MNIST typically outperforms CIFAR-10 in terms of accuracy and computational efficiency. However, it's worth noting that the relationship between testing accuracy and batch size is confirmed, while the data doesn't support the hypothesis that CNN performance hinges on batch size.

**PART VI: Improve Convolutional Neural Network Performance (20 Points)**

**Redesign CNN to improve accuracy:**

To enhance the performance of the CNN, an additional convoluted layer has been incorporated on top of the existing two layers. However, after adding the pooling layer, there was no significant improvement observed in the accuracy.

A screenshot of a computer

Description automatically generated

Despite variations in the data and the batches used for processing, the ultimate accuracy achieved was **67.81%.**

A screenshot of a computer

Description automatically generated

**PART VII: Project Report (20 Points)**

**Summary**

This deep learning project explored advanced techniques in artificial intelligence. It aimed to understand complex data, make predictions, and push the limits of machine learning using sophisticated neural networks.

* **Collecting and Preparing Data:** The project put a lot of effort into gathering high-quality data that was relevant to the problem. It also involved preparing the data, which meant making sure it was consistent, removing any unwanted information, and getting it ready for training deep learning models.
* **Choosing Neural Network Designs:** The project included creating and using neural network designs that were tailored to solve the defined problem. These designs could be Convolutional Neural Networks (CNNs) for images, Recurrent Neural Networks (RNNs) for sequences, or Transformer models for understanding language.
* **Training and Evaluating Models**: The project involved rigorous training and evaluation of models. It used powerful technology and software tools to analyze how well the models performed. Fine-tuning various settings and parameters was important to get the best possible results.
* **Putting Models to Work:** To make the project's work practical, it explored how to use the trained models in real-life situations. This meant making AI systems more accessible and useful, whether in automated processes, recommending things, or creating intelligent
* **The project's roadmap encompasses several key stages:** data collection, importing libraries, data pre-processing, CNN construction, data augmentation, network training, and accuracy assessment.
* The most crucial phase of network development involves convolution, pooling, and flattening. These operations are pivotal for extracting image features, reducing data size, and transforming the output into a format suitable for further processing.
* Convolution's main role is to detect essential features within the input image by applying small data blocks while preserving the spatial connections between pixels.
* The outcome of the convolution process is known as the "feature map." After each convolution operation, a rectified linear unit (ReLU) operation is applied to this feature map. Following this, the next operation is pooling.
* Pooling serves to reduce the dimensionality of each feature map while preserving critical information.
* In the fourth part of the project, we constructed the CIFAR-10 model, which included two CNN layers and two pooling layers designed to reduce dimensionality. Our analysis revealed that the highest achievable accuracy reached 60%.
* As we processed batches, the initial accuracy started at around 10% and gradually rose to 65% after completing 4900 steps. In contrast, using the same architecture with MNIST data, we achieved nearly 99% accuracy with CIFAR-10 data. To enhance the model's performance, we explored options such as adding more CNN pooling layers and increasing batch sizes.
* Part V involved comparing the accuracy of MNIST and CIFAR-10 data, while in part VI, we concentrated on improving the model's accuracy. Although we made a minor architectural adjustment by introducing an additional CNN layer and pooling layer, we did not observe any accuracy improvement.

**Conclusion:**

The deep learning project served as a testament to the power of artificial intelligence and its wide-ranging applications. It showcased the potential of neural networks to tackle complex problems and highlighted the need for continuous innovation in this dynamic field.

* The experiments suggest that the model's classification accuracy is influenced by the dataset's unique features.
* Achieving accurate data classification requires tailoring the model to match the dataset's specific characteristics.
* In this instance, both the CNN model used and the architecture of the MNIST dataset are similar. However, the MNIST dataset outperforms the CIFAR-10 dataset in terms of accuracy.
* It's essential to recognize that these two experiments alone do not provide enough evidence to definitively determine the effectiveness of adding a CNN layer. The results indicate that incorporating a CNN layer does not enhance model performance, but it does increase computational time.