ImageNet Classifier Comparison: Assignment 3
Group 17

Name	**Student Number (SN)**	**Contribution**
Karan Goel	7836685	100%
Alvin Jose	8066358	100%
Ashutosh Bhosale	7795786	50%
Banin Sensha Shreshta	8447196	100%
Gaurav Adarsh Santosh	7032663	100%
Lino Thankachan	7926017	100%
Rishab Manokaran	7863974	100%

CSCI946 Big Data Analytics Assignment 3
October 28, 2024

1 Introduction

In this assignment, we analyzed the pretrained features extracted from a large image recognition model and developed a classifier to achieve high classification accuracy on two provided Imagenet test datasets.

To accomplish this, we will follow a structured approach outlined in Tasks 1-4.

Task 1 involves designing a comprehensive big data analytics project while adhering to the principles of the Big Data Analytics Lifecycle.

Task 2 will focus on detailing the methodology used in our analysis and classification processes.

Task 3 will discuss the results discovered in our analysis and classification processes.

Task 4 will summarize our conclusions and the insights gained from completing this assignment.

2 Task 1: Data Analysis Design

To conduct large-scale image analysis, our first step involved employing the Big Data Analytics Lifecycle. This structured approach ensures a thorough examination of data, facilitating informed decision-making throughout the project.

2.1 Domain

The primary objective of this project was to develop a robust classifier that utilizes pretrained features to achieve high accuracy on two distinct ImageNet test datasets: ImageNet and ImageNetV2. This necessitated a comprehensive examination of the datasets to understand the pretrained features and identify the most suitable algorithms for our classification problem.

Dataset

The datasets consist of features obtained from a large language model trained on the ImageNet and ImageNetV2-MatchedFrequency datasets.

- **ImageNet**: A foundational dataset in the field of computer vision, ImageNet comprises over 14 million images distributed across 1,000 classes. This diversity and scale make it an invaluable resource for training machine learning models.
- **ImageNetV2**: Developed as a successor to ImageNet, ImageNetV2 serves as a benchmarking tool for models initially trained on ImageNet. It contains a revised set of images that are more representative of real-world distributions.
- **ImageNetV2-MatchedFrequency**: This dataset is a tailored version of ImageNetV2, where the distribution of images is adjusted to reflect the frequency of classes as they appear in natural environments. This adjustment aims to reduce the bias found in the original ImageNet dataset, providing a more balanced representation of classes.

In our analysis, we utilized ImageNet features to train our model while leveraging the ImageNetV2-MatchedFrequency dataset for validation purposes.

Resources & Goals

Resources: Given the substantial size of the datasets, we opted for a high-capacity machine, specifically an A100 instance available in Google Cloud, equipped with 80GB of RAM and 40GB of GPU memory. This configuration allowed us to conduct our experiments efficiently without significant computational constraints.

During the initial phase, we formulated key goals and posed critical questions to guide our analysis:

Key Questions:

- How do the differences between the datasets affect the classifier's accuracy? Will these differences impact precision as well?
- To what extent does preprocessing the features influence the model's performance, especially given that they already exist in a high-dimensional space?

2.2 Framing the Problem & Initial Hypotheses

Problem Type

The primary challenge of this project is **fine-tuning** and **classification**. We require a model capable of handling large datasets effectively while being trainable within our resource constraints.

Hypotheses

Next, we developed a set of structured hypotheses to guide our approach throughout the project:

- **Null Hypothesis (H0)**: There is no significant difference in classification accuracy between the two test datasets when utilizing ImageNet pretrained features.
- **Alternative Hypothesis (H1)**: A significant difference in accuracy exists between the two datasets.
- **Additional Hypotheses**:
  - **H0**: The pretrained features extracted from ImageNet provide comparable classification performance to those from ImageNetV2.
  - **H1**: There is a distinct difference in classification performance between the features extracted from ImageNet and those from ImageNetV2.

These hypotheses will inform the model testing and analysis, particularly in evaluating how variations in datasets impact performance.

2.3 Data Preparation

Although we did not focus extensively on data processing due to the availability of high-dimensional features, we organized the data in a pandas DataFrame and converted it to tensors for GPU training. In later phases, we conducted experiments incorporating normalization layers to assess their impact on model performance.

2.4 Data Split

The training dataset was utilized as is, with the model trained in batches. Subsequently, we tested the model on the designated test features.

2.5 Visualizing Data

To better understand the datasets, we employed various visualization techniques—including correlation matrices, line charts, and bar plots. These visualizations allowed us to examine dataset distributions, feature relationships, and identify any potential inconsistencies.

2.6 Model Selection

Given the pre-extracted features from pretrained models, we concluded that a neural network would be the most suitable approach for our task. Although traditional machine learning algorithms such as K-Nearest Neighbors and Decision Trees were considered, their training demands, especially concerning our limited resources, led us to favor neural networks.

2.7 Training and Testing

Our model was trained over 100 epochs, utilizing a batch size of 65,536 and a learning rate of 5×10^{-5}. We also incorporated techniques such as weight decay for regularization to enhance model robustness.

For tracking our experiments, we utilized Weights & Biases, a platform that facilitates comprehensive monitoring and management of machine learning experiments.

2.8 Final Deliverables

The final evaluation of the models included several key metrics to provide a comprehensive view of the classifier's performance:

- **Accuracy**: Indicates the proportion of correctly classified images, serving as a straightforward measure of performance.
- **Precision**: Offers insights into the true positive rate, particularly crucial in scenarios where dataset distributions are unbalanced.
- **Loss Curves**: These curves are essential for neural network training, allowing us to track the training process and identify issues related to overfitting or underfitting.

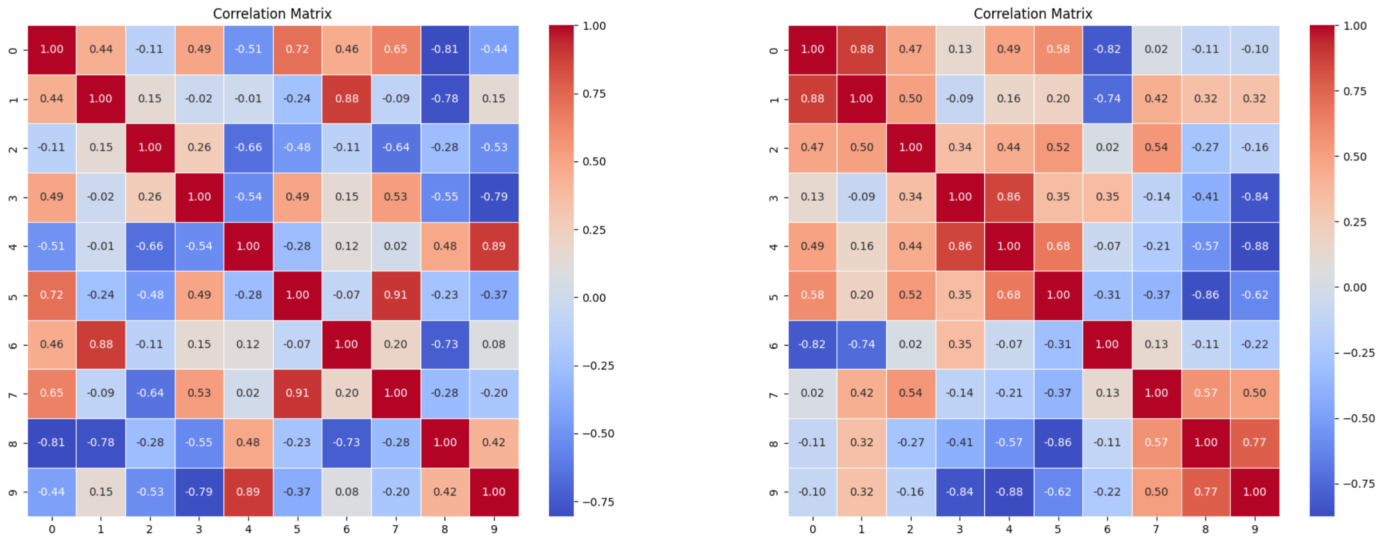
3 Task 2: Methodology

3.1 Feature Extraction Setup

For this project, pretrained features were provided. To better understand their structure before selecting a specific methodology, we first examined the extracted features prior to the final layers. This provided us with valuable insights for designing our approach.

Next, we visualized the provided features in both test datasets by plotting their correlation matrices, as shown in Fig. 1.

While individual correlation matrices offered limited information in isolation, comparing them revealed shared feature patterns, with differences in frequency distributions. This observation aligns with expectations, given that the ImageNet V2 dataset is derived from ImageNet, as illustrated in Fig. 2.



(a) Feature Correlation for Test Dataset 1

(b) Feature Correlation for Test Dataset V2

Figure 1: Correlation Matrices for Datasets V1 and V2

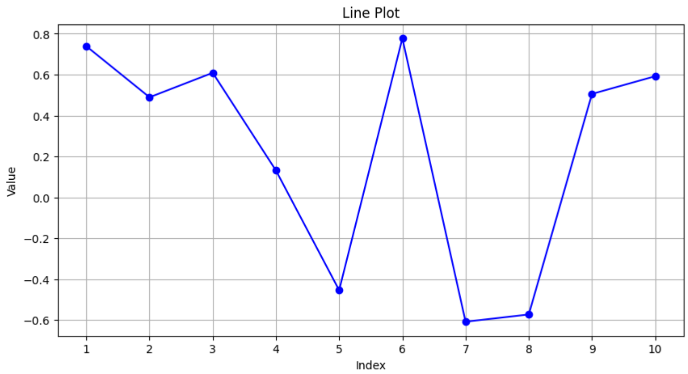


Figure 2: Comparison of Feature Frequencies

3.2 Neural Network Configurations

We designed and analyzed two neural network architectures: a simple Feedforward Network and a Batch Normalized Network. The Feedforward Network serves as a baseline model, while the Batch Normalized Network includes normalization and dropout layers to evaluate their impact on performance and generalization.

3.2.1 Feedforward Network

The Feedforward Network (denoted as NN1) consists of:

```
NN1(
  (fc1): Linear(in_features=1024, out_features=512, bias=True)
  (relu): ReLU()
  (fc2): Linear(in_features=512, out_features=1000, bias=True)
)
```

This architecture includes:

- **Input Layer**: Accepts 1024-dimensional feature vectors.
- **Hidden Layer**: A fully connected layer with 512 neurons, followed by a ReLU activation function.

- **Output Layer**: A fully connected layer producing 1000 class outputs.

3.2.2 Batch Normalized Network

The Batch Normalized Network (denoted as NN2) is structured as follows:

```
NN2(
  (fc1): Linear(in_features=1024, out_features=1024, bias=True)
  (bn1): BatchNorm1d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc2): Linear(in_features=1024, out_features=512, bias=True)
  (bn2): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc3): Linear(in_features=512, out_features=256, bias=True)
  (bn3): BatchNorm1d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc4): Linear(in_features=256, out_features=1000, bias=True)
  (dropout): Dropout(p=0.3, inplace=False)
)
```

Key components include:

- **Input Layer**: Processes 1024-dimensional feature vectors.
- **Hidden Layers**:
  - First fully connected layer (1024 neurons) followed by batch normalization.
  - Second fully connected layer (512 neurons) with batch normalization.
  - Third fully connected layer (256 neurons) with batch normalization.
- **Output Layer**: Produces outputs for 1000 classes.
- **Dropout Layer**: Applied with a dropout probability of 0.3 to mitigate overfitting by randomly omitting a fraction of neurons during training.

Incorporating batch normalization and dropout layers is expected to enhance model generalization and reduce overfitting.

3.3 Hyperparameter Tuning

Hyperparameter tuning was conducted using Weights and Biases, allowing us to iteratively record and optimize model performance. We experimented with learning rates, optimizers, loss functions, and regularization decay rates. Detailed results of this tuning process are provided in Section 4.

3.4 Model Sharing via Hugging Face Hub

To streamline collaborative model optimization, we also integrated AI operations practices, storing our models on Hugging Face Hub. This enabled efficient accessibility and version control across the team.

4 Task 3: Results

This section presents a detailed evaluation of the model performance after fine-tuning, comparing the Feedforward Network and the Normalized Network across various performance metrics, including accuracy, precision, and loss. These metrics provide insight into each model's classification effectiveness and generalization on different datasets.

4.1 Evaluation of Key Metrics: Accuracy, Precision, and Loss

Both models yielded competitive results, demonstrating strong training and testing accuracy and precision across datasets. The tables below summarize the performance metrics achieved by each network on the test datasets.

Feedforward Network Performance

Metric	Value
Test Set 1 - Accuracy	0.89556
Test Set 1 - Precision	0.89902
Test Set V2 - Accuracy	0.8195
Test Set V2 - Precision	0.83543
Train Accuracy	0.9457
Train Precision	0.97402

Table 1: Performance Metrics for the Feedforward Network

Normalized Network Performance

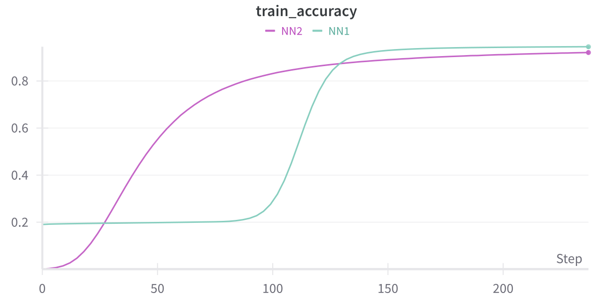
Metric	Value
Test Set 1 - Accuracy	0.88788
Test Set 1 - Precision	0.89454
Test Set V2 - Accuracy	0.8136
Test Set V2 - Precision	0.82879
Train Accuracy	0.92099
Train Precision	0.96778

Table 2: Performance Metrics for the Normalized Network

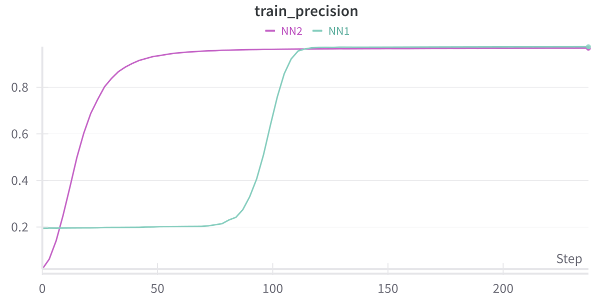
In addition to the numerical metrics, visualizations of model performance over epochs are provided below, illustrating the training process in terms of accuracy, precision, and loss evolution as the models learned.

The charts provide a visual summary of model performance throughout the training process, showing how both networks learned effectively. Notably, the Normalized Network exhibited better initial performance, while the Feedforward Network gradually caught up over time.

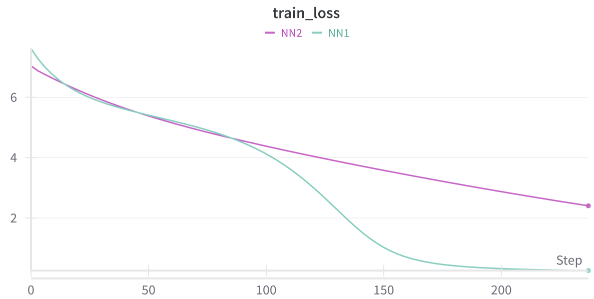
The performance difference between the V1 and V2 datasets, illustrated in Figures 4 and 5, confirms a significant 10-12% gap, supporting hypothesis H1. While generalization improved early model accuracy, the models were not able to entirely bridge the gap relative to the ImageNet-Matched Frequency dataset.



(a) Training Accuracy for Train Dataset

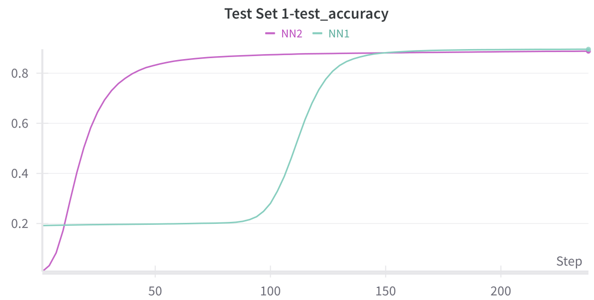


(b) Training Precision for Train Dataset

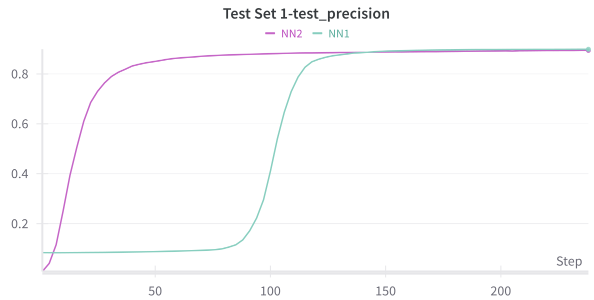


(c) Training Loss for Train Dataset

Figure 3: Performance Metrics for Train Dataset



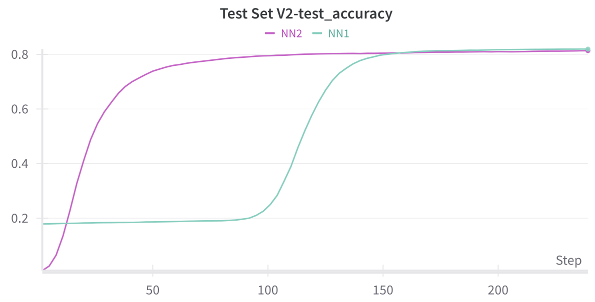
(a) Training Accuracy for V1 Dataset



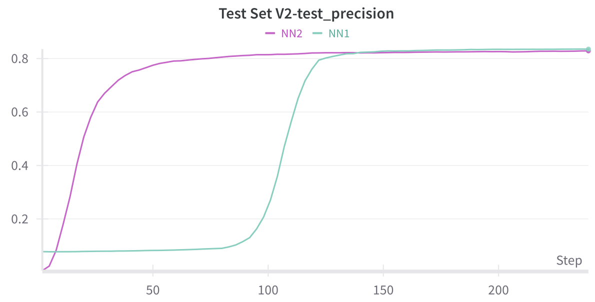
(b) Precision for V1 Dataset

(c) Loss for V1 Dataset

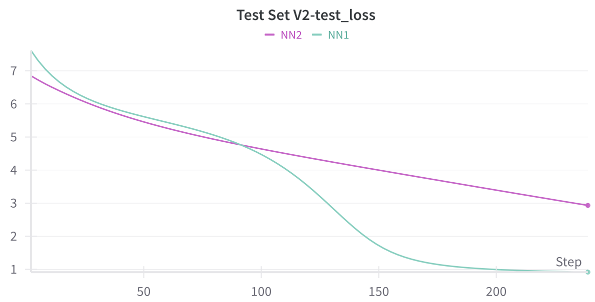
Figure 4: Performance Metrics for V1 Dataset



(a) Accuracy for V2 Dataset



(b) Precision for V2 Dataset



(c) Training Loss for V2 Dataset

Figure 5: Performance Metrics for V2 Dataset

5 Task 5: Conclusion

In this project, we explored the efficacy of using pretrained features to build an effective classifier for high-dimensional image datasets, specifically focusing on ImageNet and its successor, ImageNetV2-MatchedFrequency. By employing the Big Data Analytics Lifecycle and a structured approach, we achieved high classification accuracy, addressing the challenge of real-world data variability. Our results indicate that leveraging pretrained features can yield robust models with strong generalization capabilities, especially when equipped with architecture adjustments such as batch normalization and dropout.

Our analysis revealed key insights into the differences between datasets: ImageNetV2, with its adjusted class frequencies, presented a distinct challenge that highlighted the importance of dataset alignment when assessing model performance. The neural network models we developed showed consistent performance across both datasets, affirming our hypotheses on the utility of pretrained features. Moreover, our comparative exploration between the basic feedforward and batch-normalized networks underscored the advantage of normalization layers in managing large datasets and improving accuracy.

In summary, this project reinforces the role of pretrained features and structured neural architectures in handling complex classification tasks. It also emphasizes the need for careful consideration of dataset characteristics when fine-tuning models for real-world applications, paving the way for future research on optimizing classifiers for diverse datasets and advancing the application of big data analytics in image recognition.

References

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, *ImageNet: A large-scale hierarchical image database*, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009. Available: `https://image-net.org`

A. Paszke, S. Gross, S. Chintala, et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, NeurIPS 2019. Documentation available: `https://pytorch.org/docs/stable/index.html`

M. Abadi, P. Barham, J. Chen, et al., *TensorFlow: A system for large-scale machine learning*, OSDI 2016. Documentation available: `https://www.tensorflow.org/`

Hugging Face Inc., *Transformers Documentation*, Available: `https://huggingface.co/docs/transformers/index`

Weights and Biases, *Experiment Tracking with W&B*, Available: `https://docs.wandb.ai/`