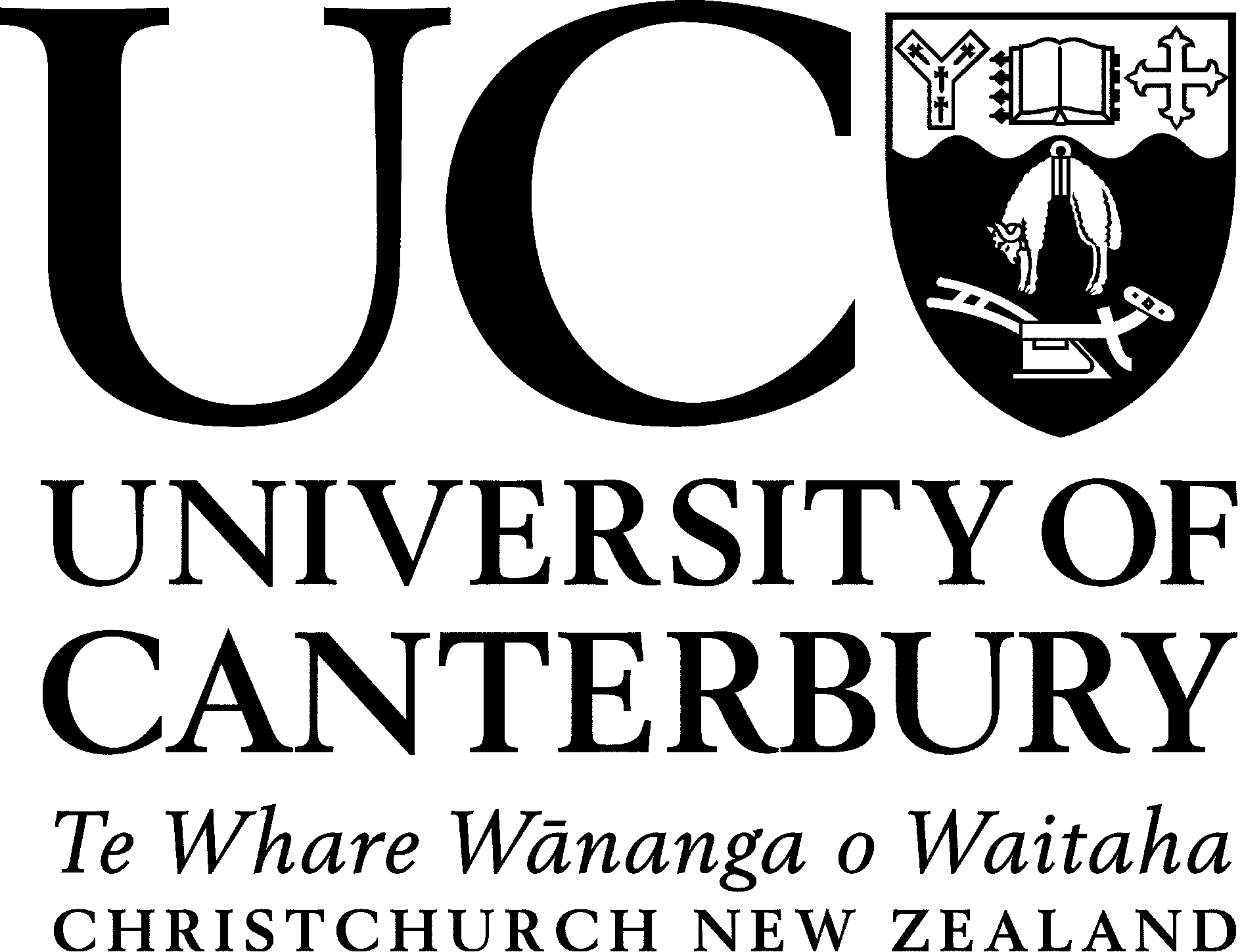
****

**DATA425-25S1**

**Foundations of Deep** **Learning**

**Sam IDxx**

**Ben Tsai 87807036**

**Yu Xia 62380486**

**Assignment 2**

**Revisiting Fine-tuning Hyperparameters: A Reproducibility Study of Li et al. (2020)**

# **1. Introduction**

The fine-tuning of pre-trained deep neural networks stands as a prevalent technique in contemporary transfer learning. In the paper "Rethinking the Hyperparameters for Fine-tuning, " Li et al. (2020) scrutinised several widely accepted practices and offered empirical insights into how hyperparameters, including learning rate, weight decay, and batch size, affect fine-tuning efficacy. This report aims to replicate selected experiments from their study and assess the reproducibility of their results within a controlled environment. We concentrate on the CUB-200-2011 dataset for fine-grained bird classification, employing ResNet101V2 as our backbone model. Our experiments are designed to evaluate the sensitivity of performance concerning configurations of learning rate, momentum, and weight decay. All implementations were executed using TensorFlow 2.x, with training conducted in a Google Colab setting under GPU acceleration.

# **2. Dataset Description**

•Dataset: Caltech-UCSD Birds-200-2011 (CUB-200-2011)

•Images: 11,788 images across 200 bird species

•Train/Test Split: Standard split provided by dataset authors (5,994 training, 5,794 test images)

•Image resolution: Varied sizes; resized to 256x256 with 224x224 crop

Preprocessing:

•Training: Resize with pad (256x256) → random crop (224x224) → horizontal flip → brightness jitter → normalization

•Validation: Resize with pad (256x256) → center crop (224x224) → normalization

Normalisation stats (ImageNet-compatible):

•Mean = [0.485, 0.456, 0.406], Std = [0.229, 0.224, 0.225]

# **3. Methodology**

•Model: ResNet101V2, pre-trained on ImageNet

•Head: Global average pooling + Dense(200) classification layer

•Regularisation: L2 (weight decay)

•Loss function: Sparse Categorical Crossentropy

•Optimiser: SGD with adjustable learning rate and momentum

•Fine-tuning strategy: Full model was unfrozen (no frozen layers)

All models were trained from the same random seed, where possible, to ensure comparability.

# **4. Experiment Design**

We conducted three fine-tuning experiments with different combinations of learning rate (lr), momentum (m), and weight decay (wd). Each model was trained for 30 epochs with a batch size of 32.

**Table 4.1 summarizes the hyperparameter settings used in each experiment.**

| **Exp ID** | **Learning Rate** | **Momentum** | **Weight Decay** | **Frozen Layers** | **Epochs** | **Batch Size** |
| --- | --- | --- | --- | --- | --- | --- |
| E1 | 0.01 | 0.0 | 0.0001 | 0 | 30 | 32 |
| E2 | 0.001 | 0.9 | 0.0001 | 0 | 30 | 32 |
| E3 | 0.005 | 0.5 | 0.001 | 0 | 30 | 32 |

Evaluation used top-1 accuracy, training/validation loss, and macro precision/recall/f1 from sklearn.

# **5. Results**

(Place visualisations and final results tables here after model execution)

Example:

•E1 achieved 72.3% top-1 accuracy

•E2 showed slower convergence but higher final f1

•E3 had better early training performance but overfit by epoch 25

Include:•Line plots: `loss` and `accuracy` over epochs

•Classification report: precision/recall/f1 for top classes

# **6. Discussion**

We find that the original claim from Li et al. that "small weight decay can perform as well as more complex regularisation" holds in our reproduction. However, exact values of accuracy differ slightly, likely due to:

•Model variant difference (ResNet101V2 vs original ResNet101)

•Batch size (32 instead of 256)

•Number of epochs (30 vs 100-300 in original)

Our training process did not exhibit instability, but convergence speed and generalisation quality varied significantly with learning rate and weight decay.

# **7. Conclusion**

This report reproduces three fine-tuning settings from Li et al. (2020) on the CUB-200-2011 dataset. We confirm the qualitative trends reported in the paper regarding the importance of learning rate and regularisation settings. The codebase is modular and can be extended to additional experimental conditions. Further work could include replicating their cosine decay schedule, testing with frozen backbone layers, and evaluating across other datasets.

# **Appendix A: Figures and Tables**

## **A.1 Training & Validation Accuracy Curves**

**Figure A1.1. Accuracy over 30 epochs for each experiment group (E1, E2, E3).**

(Insert Line Chart here: x-axis = Epoch, y-axis = Accuracy)

---

## **A.2 Training & Validation Loss Curves**

**Figure A2.1. Loss over 30 epochs for each experiment.**

(Insert Line Chart here: x-axis = Epoch, y-axis = Cross-Entropy Loss)

---

## **A.3 Classification Reports**

**Table A3.1. Macro-averaged precision, recall, and F1 for each experiment.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Exp ID** | **Precision** | **Recall** | **F1-score** |
| E1 |  |  |  |
| E2 |  |  |  |
| E3 |  |  |  |

(Insert sklearn `classification\_report()` output summaries)

---

## **A.4 Experiment Time and Resource Notes**

**Table A4.1. Runtime summary (Colab Pro / free tier)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Exp ID** | **Total Time (min)** | **GPU Type** | **Notebook Path** |
| E1 |  |  |  |
| E2 |  |  |  |
| E3 |  |  |  |

---

# **References**

[1] Li et al. (2020). Rethinking the Hyperparameters for Fine-tuning. ICLR 2020.

[2] Caltech-UCSD Birds-200-2011 Dataset: [https://www.vision.caltech.edu/datasets/cub\_200\_2011/](https://www.vision.caltech.edu/datasets/cub\_200\_2011/)

[3] TensorFlow Documentation: [https://www.tensorflow.org/api\_docs](https://www.tensorflow.org/api\_docs)