Lab Assignment 2

1 Overview

Welcome to the second lab assignment for Neural Networks and Deep Learning! Please be sure to **read this document in its entirety**. In this assignment, you will scientifically investigate how various regularization techniques influence model *data efficiency*, meaning how many training samples are required in order to obtain models with good generalization. Your investigation will involve two primary types of independent variables: (1) the amount of training data used and (2) the regularization technique. For the latter, you will explore two distinct types: architectural regularization and data/loss regularization. Along the way, you will practice using common "building blocks" to construct complex deep models. This assignment is worth a total of **50 points**, and an extra credit opportunity is provided which is worth **5 points**.

2 Selecting a Dataset

Choose a real-world dataset not covered in the course that is designed for the multi-class classification problem. You can use datasets from any existing library (e.g. PyTorch, Tensorflow), website (e.g. Kaggle, Hugging Face), or even use your own data! The only requirement is that the task is multi-class classification, and the dataset contains at least 1000 total samples. The programming tutorials are a good place to look for inspiration. Pre-processing the data is highly recommended, but not required. If the dataset you choose does not already come with a train/validation/test split, then divide the dataset into a 70/30 train/test split.

3 Designing a Baseline Model

To conduct meaningful experiments, you will need a solid baseline to compare to. To accomplish this, design a classification model that works with your chosen dataset. Existing architectures are **NOT PERMITTED** for this assignment. Instead, you must **construct your own model**. The model must contain at least 5 non-linear activation layers, with at least 1 learned layer in between each of them. You are permitted to use existing "building blocks", e.g., Conv2d(), Linear(), Pool(). However, dropout and batch normalization layers are **NOT** allowed in the baseline model. You may use any automatic differentiation packages you like. You are **NOT** required to implement the back-propagation for the model.

Programming tutorial 4 is a good reference for how to construct models from layer modules using PyTorch. It is recommended to use layers which make sense for your chosen dataset, for example: use 2-D convolution layers for an image dataset and recurrent layers for sequence data. This baseline model will form the basis for all experiments in this lab, so you are encouraged to play around with module-level model design before finalizing your architecture.

3.1 Evaluating the Baseline Model

Train your baseline model using the cross-entropy loss with the SGD optimizer long enough that it converges well. The number of epochs required for convergence will vary depending on your dataset and model, so you may need to try a few different options. Choose the fewest number of epochs that results in reasonably good performance. What constitutes "good performance" will depend on your dataset, so you may need to look up how well established models perform. Your architecture does **NOT** need to match the performance of advanced models, but it should be trained long enough that you can be confident it has learned all that it can. Fix this number of epochs (we will denote it by E) and all other hyper-parameters (e.g., batch size) for all other experiments. Record the performance of your baseline model when trained for E epochs.

To measure the baseline data efficiency, retrain your model using 25%, 50%, and 75% of the total training data. These percentages stack multiplicatively with the train/test split, i.e., 50% of the training data refers to half of the train split, not half of the total dataset. When sub-sampling the training data, be sure to retain the original class distribution. E.g., for 50% of the training data you should select half of the training samples from each class. Record the performance of the baseline model in each of these cases.

4 Experiment 1: Architectural Regularization

For your first experiment, investigate how modifications to your baseline architecture influence data efficiency. Choose **one** regularization technique from the below list:

1. Dropout layers
2. Residual/skip connections
3. Batch normalization layers

Next, modify your baseline model by augmenting it with your chosen regularization technique. For example, add dropout layers after the non-linear activation layers. Train this augmented model for E epochs on 25%, 50%, 75%, and 100% of the training data. Use *exactly* the same hyper-parameters you used to train the baseline model. Record the results.

5 Experiment 2: Data/Loss Regularization

For your second experiment, investigate how other types of regularization influence data efficiency. Choose **two** regularization techniques from the below list:

1. L1 or L2 weight penalties
2. Data augmentation
3. Alternative optimizers: E.g., Adam, SGD w/ momentum, Adagrad
4. Learning rate scheduling: E.g., step decay, exponential decay

Retrain your baseline method with your chosen regularization techniques. Train these regularized models for E epochs on 25%, 50%, 75%, and 100% of the training data. Use *exactly* the same hyper-parameters you used to train the baseline model. Record the results.

6 Deliverables

Prepare a 2-3 page report discussing your experimental designs, results, and analysis for sections 4 and 5. This report should be well formatted, clearly understandable, and convey what you found. Details on each section of the report are discussed in the next sections.

6.1 Experimental Design

The first section of your report should discuss *what* your experiments aim to study (e.g. which regularization techniques are you using?), *how* it will be conducted (e.g., which variables will be changed? Which left fixed as controls?), and *what* you expect to find (hypothesized outcome). You **MUST** report all architectural and training details (including hyper-parameters!) used in your study. This section should include a discussion of how the baseline architecture was designed. When writing this section, ask yourself "could someone else reproduce this experiment solely based on the report?" If the answer is no, then you need to add detail. You should convey the experimental design in words; copy-pasting code is **NOT** acceptable for this section.

6.2 Results

The second section of your report should provide the results of your experiments. This can be accomplished with plots, tables, charts, etc. Your figures **MUST** be clearly formatted, well annotated, and understandable by anyone with a baseline knowledge of deep learning. Remember to include labels for axes, titles/captions for figures, legends, and any other information required for the reader to understand your results. There should be *zero* ambiguity about what information is provided by the figures. You are free to decide how to format the results, but there are a few requirements. First, report the test accuracy for the baseline, each regularization technique, and each amount of training data. Second, report the average recall and precision (across all classes) for each configuration. Below are some result formatting ideas for inspiration:

1. Tables. List the amount of training data used along the columns, and the regularization technique used across the rows.

2. Plots. Display the amount of training data used on the x-axis, with test accuracy on the y-axis. Use different color lines for each regularization technique and provide a legend.

3. Bar Charts. Set the bar height according to the test accuracy, and color-code based on regularization technique and amount of training data used.

6.3 Analysis

The third and final section of your report should discuss what you learned from your experiments. This should be more than a superficial reiteration of what can be observed in your results. Do **NOT** simply state what the results are; rather discuss *what the results say*. What trends did you observe from the experiments? Which regularization techniques improved data efficiency the most? Is there an optimal trade-off point between

amount of training data and model performance? Why do you think the observed trends occur? You should discuss in depth at least *2 trends* in this section.

6.4 Submission Format

Please submit a pdf named with your first and last name: firstname_lastname.pdf. A successful submission will consist of two self-contained, separate contributions. First, it will include a report with all methods, results, and analysis (portions indicated by "Report") **as the first part of the PDF file**, each broken out into a separate section. Second, it will include the source code of your implementation **as the second part of the PDF file** (portions indicated by "Code").[1] We will only review the code when the report does not contain sufficient detail, in order to provide partial credit. To avoid this **automatic deduction**, please ensure your report properly conveys three pieces of information: (1) *how did you conduct your experiments*, (2) *what results you obtained from the experiments*, and (3) *what the results revealed about regularization and data efficiency*.

6.5 Extra Credit: 5 Points

For this extra credit assignment, investigate how model depth relates to regularization techniques. To start, modify your baseline model to contain at least 15 non-linear activation layers, with at least 1 learned layer in between each of them. Train this new baseline model for E epochs using the same hyper-parameters as the other experiments. Then, train the baseline deep model with only 50% of the training data. Record the results.

Next, choose the **single best** performing regularization technique across experiments 1 and 2 combined. You should select only 1 of the 3 techniques. Train the baseline deep model with your chosen technique using 50% and 100% of the training data. What patterns do you observe in the results? Provide a short ($\sim$ 0.5 page) discussion of what you learned from this experiment. This does not need to be a full report, but should at a minimum convey: (1) how your deep model was constructed, (2) which regularization technique was chosen, and (3) how the chosen regularization technique and amount of training data influence the performance of the deep model. Which findings match and which differ from the experiments with the smaller models?

Collaboration versus Academic Misconduct: Collaboration with other students and AI is permitted, but the work you submit must be your own. Copying/plagiarizing work from another student or AI is not permitted and is considered academic misconduct. For more information about University of Colorado Boulder's Honor Code and academic misconduct, please visit the course syllabus.

[1] We require submitting the code as a PDF