Stellenbosch
UNIVERSITY
IYUNIVESITHI
UNIVERSITEIT

Computer Science Division

Machine Learning
CS441/741

Assignment 3: Neural Networks

Due: 26 September 2025, 08:00

Instructions

For this assignment, we are going to park the forest cover type dataset. You are going to implement and evaluate some neural networks, and for this purpose, the forest cover type dataset will result in too long training times.. For the purposes of this assignment, please note the following:

- You may select any one of the options listed below.
- Please submit your own work.
- You may program in any language, provided that your code compiles and execute on Linux. While you are allowed to make use of machine learning libraries, note that you learn most about the machine learning algorithms when you implement these algorithms yourself.
- You have to write a report, using the IEEE conference template (google!), in two-column, 10pt format. Please see assignment 2 for tips on report writing. Also see the writing rules – uploaded to your STEMlearn module – for a list of writing rules. Please consult these writing rules and apply them. Note that only the report will be evaluated, not your code. Therefore, a bad report will result in bad marks.
- Submit your report in pdf format. Note that no format other than pdf will be accepted. Make sure that your report has a reference to your git repository for your code. Code will not be evaluated, but may be scrutinized if found necessary. Provide a link to the Git repository that contains your code. Make sure that you name your report file as `????????RWxxxassignment3.pdf`, where you replace the question marks with your student number, and `xxx` with the module code that you are registered for. Please note that I use a script to pull out all reports, and if you do not follow this file naming convention, your report **will not** be extracted for evaluation.

- Make sure that your name, surname and student number are clearly indicated in the front matter (title section) of your report. If there is no identification of the author of the report in the front matter of your report, your report **will not** be evaluated.
- Generative AI tools may not be used.
- Upload your report via STEMlearn before the deadline of **26 September 2025, 08:00**. Note that late assignments will not be accepted. After this deadline, I will extract all reports to start with evaluation of the reports that day.

Option 1: Mini-Batch Training using Dynamic Meta-Heuristics

Consider this assignment only if you have done AI791 (Artificial Intelligence).

Recent research on fitness landscape analysis of the error surfaces produced by mini-batch training has shown that the error surfaces change over time (see `NNFLA.pdf` available on SUNLearn). Neural network training using mini-batches is therefore essentially a dynamic optimization problem. It is therefore hypothesized that it is best to make use of an optimization algorithm developed to solve dynamic optimization problems to train a neural network where mini-batching is employed. For this assignment, you are going to test this hypothesis.

For the purposes of this assignment, note the following:

- Select three classification problems. In your report, describe (under the appropriate section) these problems and also describe any data preprocessing that you had to do, with proper motivations.
- Decide on five different mini-batch sizes ranging from small to large. Remember to provide detail on the chosen mini-batch sizes in the appropriate section of your report.
- Use only one hidden layer, with an overestimate of the number of hidden units. In order to prevent overfitting, make use of a regularization approach such as weight decay. For this, you also have to consider that the penalty coefficient is problem dependent. Discuss the regularization approach and the approach that you have use to set the value of the penalty coefficient in your report. Make sure to provide detail on the activation functions used.
- Carefully decide on the performance measures to use to quantify the quality of the trained neural networks.
- Evaluate the performance of stochastic gradient descent for the different mini-batch sizes on the different classification problems. Make sure to discuss all control parameters and how you have decided on their values.
- Now, select a meta-heuristic that was developed to solve dynamic optimization problems, e.g. the quantum-inspired particle swarm optimization algorithm or the DynDE algorithm. Then evaluate the performance of this meta-heuristic as training algorithm for the different mini-batches on the different classification problems. Make sure to discuss all control parameters and how you have decided on their values.
- Also implement a standard version of the meta-heuristic, developed to solve static optimization problems, such as the standard inertia particle swarm optimizer. This is added as a control algorithm to see if the dynamic meta-heuristic actually contributed to differences in performance.
- Determine which of the approaches performed best, and provide an outcome to the stated hypothesis. Provide your detailed results and a discussion thereof in your report.

Option 2: Comparison of Feedforward Neural Network Training Algorithms

For this assignment you have to compare the performance of the following three neural network training algorithms:

- Stochastic gradient descent.
- Scaled conjugate gradient (see `SCG.pdf` available on SUNLearn).

- LeapFrog (see `LFROGa.pdf` and `LFROGb.pdf` available on SUNlearn).

For the purposes of this assignment, note the following:

- Select at least three classification and three function approximation problems of varying complexity. In your report, describe (under the appropriate section) these problems and also describe any data preprocessing that you had to do, with proper motivations.
- Use only one hidden layer. For each problem, you have to determine the optimal number of hidden units. For this purpose, you need to decide what is meant by optimality in this context, and design an appropriate empirical process to find the optimal number of hidden units. Provide sufficient empirical evidence to support your findings.
- Also find the best possible values for the control parameters of the training algorithms that you compare. As for the optimal number of hidden units, design an appropriate empirical process to find best values, and sufficient empirical evidence to support your findings.
- Decide on the performance criteria that you will use to compare the performance of the training algorithms. Define an appropriate empirical process that will aid in a statistically sound approach to determine which of the three training algorithms performed best.
- You may decide on any activation function and cost function.

Option 3: Active Learning in Neural Networks

For this assignment, you have to compare two approaches to active learning with standard passive learning using stochastic gradient descent. The purpose of the assignment is to determine which of the approaches provide the best performance, and to identify the advantages of active learning compared to passive learning.

The approaches to compare are:

- Passive learning using stochastic gradient descent.
- Active learning using neural network output sensitivity analysis (see `SASLA.pdf` available on SUNLearn).
- Active learning using uncertainty sampling (see `ALUS.pdf` available on SUNLearn).

For the purposes of this assignment, note the following:

- Select at least three classification and three function approximation problems of varying complexity. In your report, describe (under the appropriate section) these problems and also describe any data preprocessing that you had to do, with proper motivations.
- Use only one hidden layer, with an overestimate of the number of hidden units. In order to prevent overfitting, make use of a regularization approach such as weight decay. For this, you also have to consider that the penalty coefficient is problem dependent. Discuss the regularization approach and the approach to set the value of the penalty coefficient in your report.
- Also find the best possible values for the control parameters of the training algorithms that you compare.
- Decide on the performance criteria that you will use to compare the performance of the training algorithms. Define an appropriate empirical process that will aid in a statistically sound approach to determine which of the three training algorithms performed best.
- You may decide on any activation function and cost function.

Option 4: Time Series Forecasting using Recurrent Neural Networks

For this assignment, you are going to implement and compare various simple recurrent neural networks for time series prediction. Do this in the following steps:

- Find any five time series datasets. Describe these datasets. Make sure to indicate if the datasets are stationary or non-stationary.

- For self-study, find out how cross-validation should be applied to time series. As a starting point, refer to `https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4`. You have to make use of an appropriate cross-validation process in your analysis of the performance of the simple recurrent neural networks. Describe the process used in your report.
- Perform the necessary data pre-processing on the selected datasets. Describe what you have done and provide justifications.
- You are going to use an Elman recurrent neural network, a Jordan recurrent neural network, and a multi-recurrent neural network. Describe these neural networks in your report. You may select any optimization algorithm and appropriate loss function. Describe what you have used in your report.
- You have to make sure that the networks do not underfit or overfit. Describe the process that you have implemented to prevent undertitting and overfitting.
- Carefully define the empirical process that you have followed, and describe this process in your report. The process has to include settings for all hyperparameters, neural network architecture, performance measures, and the process followed to determine which simple recurrent neural network performed best for each of the datasets.
- Present and discuss your results, and conclude on which simple recurrent neural network performed best.

Option 5: Cooperative Particle Swarm Optimization Training Algorithm

For this assignment, you are going to explore different particle swarm optimization (PSO) algorithms as feed-forward neural network training algorithms, specifically cooperative PSO algorithms. Do this aasignment only if you have completed AI791, or if you have time to read about PSO in your own time.

For the purposes of this assignment, note the following:

- Select at least three classification and three function approximation problems of varying complexity. In your report, describe (under the appropriate section) these problems and also describe any data preprocessing that you had to do, with proper motivations.
- Use only one hidden layer. For each problem, you have to determine the optimal number of hidden units. For this purpose, you need to decide what is meant by optimality in this context, and design an appropriate empirical process to find the optimal number of hidden units. Provide sufficient empirical evidence to support your findings.
- For the PSO algorithms, you will make use of weight decay. So, remember to add the weight decay penalty function to the error function that you minimize.
- As a baseline, implement a global best inertia weight PSO algorithm to train the feedforward neural networks. State the values that you have used for the control parameters, with justificiation for why you have used these values.
- Select any one of the approaches in the paper `randomGrouping.pdf` available on SUNlearn, remembering to use weight decay, use the selected approach to train the feedforward neural networks.
- The last two algorithms that you will implement are the merging cooperative PSO (MCPSO) and the decomposition cooperative PSO (DCPSO), as described in the paper `mCPSOdCPSO.pdf` available on SUN-learn. Use these two algorithms to train the neural networks.
- Present the results for these algorithms, compare these results, and draw a conclusion on which approach performed best.

Option 6: Incremental Class Learning

For this assignment, you are going to explore the potential of incremental class learning as an approach to address class imbalance for multi-class problems. Incremental class learning refers to training of a neural network, starting the training process on only two of the classes, then as performance on the current classes stagnate, a next class is selected. Because more complexity is added over time, it is also necessary to explore a way in which the hidden layer size will be increased when necessary. While various approaches can be employed to select the order in which classes are added to the neural network architecture, for the purposes of this assignment the order will be from least representative to most representative class, i.e. from minority class to majority class.

For the purposes of this assignment, you have to search for classification problems with more than two class labels, and where class imbalance is present. If you find problems with approximately equal class distribution, you can create class imbalance through down sampling.

As the first step, train a neural network without incremental class learning and without implementing any methods to correct for class imbalance. Note that you will have to tune the hidden layer size so that the neural network does not underfit or overfit.

Now implement your incremental class learning algorithm, where you start with the two least frequent classes. Start training a neural network with the simplest archiecture, i.e. no hidden neurons. When performance stagnates, decide if the model underfitted. If you believ that the model underfitted, and hidden layer with a hidden neuron. Repeat the training process, and proceed to add more hidden neurons while you observe underfitting. As soon as you see overfitting, or you are happy that the current classification error is acceptable, add the next output neuron for the next least representative class label. Continue this process until you have trained on all of the class labels, and you are satsfied with the performance.

Compare the performance the neural network trained from the start on all class labels with the performance of the incremental class learning approaches. Did the incremental learning approach help at all? If so, try to explain why, and if not, try to explain why not.

Remember to add all detail to your report.

Mark Rubric

Your report will be evaluated as follows:

Aspect	**Mark**
Abstract	5
Introduction	10
Background	20
Implementation	10
Empirical process	15
Results & discussion	50
Conclusions	5
References	5
Linguistic quality	20
Total	140

Note that if you did not submit your own code, the mark obtained above will be multiplied by a factor of 0.5.