Course Project

Fall 2023 - CSC 4850 and CSC 6850

General Instructions This document outlines 2 project proposals. One project for CSC 4850 and another for CSC 6850. If you would like to implement your own project, please meet with the instructor to discuss further. All projects will be done individually - no groups.

The course project has 20% weight in the final grade.

1) **Preliminary Report - 5% - Due - Oct 18, 2023**:

   Report consisting of preliminary work. Minimum 2-pages excluding References. The report is submitted as a pdf file "firstName_lastName_preliminary_project_report.pdf".

   This submission will be used to give students feedback on their report formatting and evaluate the progress of the project. The report should be in the prescribed format and must include some preliminary results. Include Tables, Figures, Table and Figure captions, Equations (if any) and References in the preliminary report in order to receive feedback regarding formatting. Include placeholder Figures and Tables if you don't have any results by the preliminary report.

2) **Final Report - 15% - Due - Dec 04, 2023**:

   For the final submissions, every student will submit 2 files:

   (i) The main report is a .pdf file "firstName_lastName_final_project_report.pdf".

   (ii) A .pdf file containing the code for the experiments. Do not include an image of the code. The code must be in text format in the PDF file. The naming convention is "firstName_lastName_final_project_code.pdf".

   The project report will be strictly 4 pages, excluding References. The References will start on the 5th page. The project report should be formatted according to the 2 column IEEE CVPR conference format. The format (Latex and Word) can be downloaded from here. Students are encouraged to use Latex for creating their report. Overleaf.com is a good online Latex editor, but other editors like Microsoft Word are acceptable as well. Here is a link to OverLeaf for the CVPR template. Additional results can be added to an Appendix after the References. These will be considered as Supplementary and are not necessary for the Project Report. The following is a guideline for report preparation. Students are strongly encouraged to stick to these guidelines. The report should have the following components, Title, Author Information, Abstract, Introduction, Models, Experiments and Results, Conclusions, References.

Abstract (5) + Introduction (10) + Models (20) + Experiments and Results (50) + Conclusions(10) + References (5) = 100pts

All the sections will be graded for quality and content. The report must have the look and feel of a IEEE CVPR conference paper – so, stick to the format. The Figures and Tables must all be labeled, adequately captioned and referred-to in the text. Previous literature must be correctly cited and referred-to in the text. Mathematical notation must be proper, and the equations (if any) must all be labeled. The Conclusions must indicate what was learned/concluded from the project – it is not a rephrasing of the Abstract. Note - There is an element of subjective evaluation as well which cannot be put into a rubric.

Note: The submitted code must be the student's own implementation and self-contained. One may rely on open source code for reference but do not submit open source code as it will be tested for plagiarism.

CSC 4850: Classification

In this project we will compare the 2 classifiers we discussed in the lectures viz., Naïve Bayes and k-NN. We will use the popular MNIST digits dataset [1] for our experiments.

1) **Naïve Bayes**: Implement a Naïve Bayes classifier where the pixel conditional probabilities $p(x|y)$ are Gaussian. Plot the mean and the variance image for each category. Evaluate the test error and plot the Confusion matrix.
2) k**-NN**: Implement the k-Nearest_Neighbor algorithm to classify MNIST data. Evaluate the test error for different values of $k \in \{1, 2, 5, 10, 100\}$. We will then compare k-NN in low dimension space. We will evaluate PCA and Autoencoder dimensionality reduction. Apply PCA dimensionality reduction to reduce to 2-dimensions and implement k-NN in the 2-dimensional space. Use the centroid of each category from the training data to plot the Voronoi region for the category. Scatter plot some of the test data to see how it aligns with the Voronoi regions. Evaluate test error in the lower dimensions for different values of of $k \in \{1, 2, 5, 10, 100\}$. Repeat the experiment using an Autoencoder to perform dimensionality reduction to 2-dimensions. Compare the Voronoi-diagrams diagrams and the clustering of the data in lower dimensions. Compare the test errors with and without dimensionality reduction and analyze your results. Plot the Confusion matrices.

CSC 6850: Semi-supervised Learning

In this project we will implement Semi-supervised learning, where we train a classifier with both labeled and unlabeled data. We will use the popular MNIST digits dataset [1] for our experiments. We will use 10,000 samples for training with 1000 examples from each class. In the experiments, consider 10 labeled samples from every class and treat the rest of the samples as unlabeled. We will evaluate our results on the unlabeled data. For the classifier we will use a 2 layer neural network with ReLU activations $[784, 200, 10]$. The input is vectorized MNIST image with 784 dimensions, the hidden layer has 200 dimensions and the output has 10 dimensions for the 10 digit categories. In addition we will also apply our algorithms to a toy dataset - two-moons Toy-dataset. Create your own Two-Moon dataset. Consider 3 labeled examples each from the two categories. Treat the rest of the data as unlabeled. Display the decision boundaries using the semi-supervised learning algorithms listed below. You can use a 2 layer network with sigmoid activations as the classifier, for e.g., $[2, 10, 2]$. The input is 2 dimensions, the hidden layer is 10-dimensions, and the output is 2 categories. We will compare 3 algorithms with the baseline supervised learning (train with only the labeled data).

1) **Baseline** Train the 2 layer neural network using only the labeled training data and evalute its performance on the unlabeled data.
2) **Entropy Minimization**: Use Entropy Minimization [2], for unlabeled data along with labeled data to train a semi-supervised learning model and report the performance on the unlabeled data.
3) **Pseudo Label**: Estimate Pseudo Labels [3], for unlabeled data along with labeled data to train a semi-supervised learning model and report the performance on the unlabeled data.
4) **Virtual Adversarial Training**: Use Virtual Adversarial Training (VAT) [4] to implement semi-supervised learning and report the performance on unlabeled data.
5) **My Pseudo Label**: Pseudo labels may be noisy. Develop your own Pseudo Label technique different from [3]. This will involve assigning pseudo labels to unlabeled data and filtering them to select the best ones for training.

References

[1] Y. LeCun and C. Cortes, “MNIST handwritten digit database,” 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[2] Y. Grandvalet and Y. Bengio, “Semi-supervised learning by entropy minimization,” in *Advances in neural information processing systems*, 2005, pp. 529–536.

[3] D.-H. Lee *et al.*, “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in *Workshop on challenges in representation learning, ICML*, vol. 3, no. 2. Atlanta, 2013, p. 896.

[4] T. Miyato, S.-i. Maeda, M. Koyama, and S. Ishii, “Virtual adversarial training: a regularization method for supervised and semi-supervised learning,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.