Semester long Project: Integrating Discriminative and Generative Models

FYS5429/9429

Spring 2026

1 Introduction and Motivation

Modern machine learning spans two complementary paradigms:

- **Discriminative models**, which learn mappings from inputs to outputs (e.g. classification and regression).
- **Generative models**, which aim to learn the underlying data distribution.

While discriminative models such as neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs) are highly effective for prediction tasks, generative models such as variational autoencoders (VAEs), Boltzmann machines, and diffusion models provide deeper insight into the structure and distribution of data.

The purpose of this project is to explore the interplay between these paradigms through theoretical analysis, implementation, and empirical evaluation.

2 Project Objectives

The main objectives of the project are:

- Develop a solid understanding of both discriminative and generative modeling approaches.
- Implement and train selected models on real datasets.
- Compare performance across models and tasks.
- Investigate how generative models can enhance discriminative tasks.
- Analyze strengths, limitations, and computational complexity.

3 Project Tasks

The project consists of three main components.

3.1 Part I: Discriminative Modeling

You should implement and analyze at least two of the following models:

- Fully connected neural networks
- Convolutional neural networks (CNNs)

- Recurrent neural networks (RNNs) for temporal data
- Autoencoders (as feature extractors)

Tasks include (you should select whether you wish to study classification or regression problems):

- Training on classification datasets (e.g. MNIST, Fashion-MNIST)
- Optional regression tasks (e.g. time series prediction)
- Evaluation using appropriate metrics (accuracy, MSE, etc.)

3.2 Part II: Generative Modeling

You should implement at least one generative model:

- Variational Autoencoder (VAE)
- Restricted Boltzmann Machine (RBM) or Boltzmann machine
- Diffusion model (optional advanced component)

Tasks include:

- Learning latent representations
- Generating new samples
- Evaluating generative quality (visual inspection, likelihood proxies)

3.3 Part III: Integration and Comparison

The key part of the project is to combine the two paradigms.

Possible directions include:

- Using latent variables from a VAE as input to a classifier
- Data augmentation using generative models
- Comparing classification performance with and without generative preprocessing
- Hybrid models combining reconstruction and prediction losses

4 Datasets

You may choose among the following datasets (please feel free to add):

- **MNIST / Fashion-MNIST**: image classification
- **CIFAR-10**: more complex image classification
- **Time series datasets**: e.g. stock prices, physical signals
- **Synthetic datasets**: for controlled experiments

You are encouraged to explore both:

- **Static data** (images, tabular)
- **Temporal data** (time series, sequences)

5 Methodology

You should:

- Clearly define model architectures
- Perform hyperparameter tuning
- Use appropriate training procedures (e.g. Adam optimizer)
- Ensure reproducibility of results

The report should include:

- Mathematical formulation of models
- Description of training procedures
- Analysis of results
- Visualization of learned representations

6 Possible Extensions

If you wish to go beyond the above you may explore:

- Diffusion models for data generation
- Energy-based models and connections to statistical physics
- Bayesian interpretations of generative models
- Application to scientific data (e.g. physics simulations)

7 Conclusion

This project provides an opportunity to bridge two central paradigms in machine learning. By combining discriminative and generative approaches, the hope is that you will gain a deeper understanding of both predictive modeling and data representation, preparing them for research and applications in modern machine learning.

8 Introduction to numerical projects

Here follows a brief recipe and recommendation on how to write a report for the project.

- Give a short description of the nature of the problem and the eventual numerical methods you have used.
- Describe the algorithm you have used and/or developed. Here you may find it convenient to use pseudocoding. In many cases you can describe the algorithm in the program itself.
- Include the source code of your program. Comment your program properly.
- If possible, try to find analytic solutions, or known limits in order to test your program when developing the code.

- Include your results either in figure form or in a table. Remember to label your results. All tables and figures should have relevant captions and labels on the axes.
- Try to evaluate the reliabilty and numerical stability/precision of your results. If possible, include a qualitative and/or quantitative discussion of the numerical stability, eventual loss of precision etc.
- Try to give an interpretation of you results in your answers to the problems.
- Critique: if possible include your comments and reflections about the exercise, whether you felt you learnt something, ideas for improvements and other thoughts you've made when solving the exercise. We wish to keep this course at the interactive level and your comments can help us improve it.
- Try to establish a practice where you log your work at the computerlab. You may find such a logbook very handy at later stages in your work, especially when you don't properly remember what a previous test version of your program did. Here you could also record the time spent on solving the exercise, various algorithms you may have tested or other topics which you feel worthy of mentioning.

Format for electronic delivery of report and programs

The preferred format for the report is a PDF file. You can also use DOC or postscript formats or as an ipython notebook file. As programming language we prefer that you choose between C/C++, Fortran2008 or Python. The following prescription should be followed when preparing the report:

- Send us an email in order to hand in your projects with a link to your GitHub/Gitlab repository.
- In your GitHub/GitLab or similar repository, please include a folder which contains selected results. These can be in the form of output from your code for a selected set of runs and input parameters.

Finally, we encourage you to collaborate. Optimal working groups consist of 2-3 students. You can then hand in a common report.