

CNN Mini Project Guideline

Submission Deadline: November 18, 2025

Weighting: 15% of total grade

Purpose and Deliverable

This mini project is designed to deepen your understanding of convolutional neural networks (CNNs) and their practical applications. You will independently select a dataset and task, implement and train a CNN model using PyTorch, and critically analyze your results.

The final deliverable is a single, self-contained **Jupyter/Colab notebook file (.ipynb)**. This notebook must include all your code, the outputs of your code cells (such as logs, metrics, and plots), and detailed explanations in Markdown cells. The goal is to create a comprehensive document that serves as both your code repository and your final report.

Project Guidelines

Your notebook should be structured to address the following areas:

1. Topic Selection & Problem Definition

- Choose an image domain and a relevant task (e.g., classification, detection, segmentation).
- Data can come from established datasets (e.g., CIFAR-10/100, Caltech-256, ImageNet) or a custom dataset you collect. Ensure the problem is substantial enough to warrant experimentation.
- In a Markdown cell, clearly define the problem you are trying to solve and why it is interesting.

2. Data Preparation

- In a Markdown cell, describe the source of your data and how you split it into training, validation, and test sets. Justify your choice of split proportions.
- In code cells, implement the data loading and preprocessing steps.
- Describe and implement any data augmentation techniques (e.g., resizing, normalization, rotations, flips) and justify why they are appropriate for your dataset and task.

3. Model Design and Implementation

- In a Markdown cell, describe your baseline CNN model architecture or the pre-trained network you are adapting (e.g., ResNet, VGG). Justify your architectural choices.

- In code cells, implement the model using PyTorch.
- Propose and implement at least **one significant variation or improvement** over the baseline. Examples include:
 - Adjusting the model's depth or width.
 - Adding/tuning dropout or batch normalization.
 - Experimenting with different activation functions, optimizers, or learning rates.
- Train your models using appropriate loss functions. Record and print key hyperparameters.

4. Evaluation and Analysis

- In code cells, evaluate your baseline model and your improved model using relevant metrics. Accuracy and loss are required. For multi-class or imbalanced datasets, you should also include precision, recall, F1-score, and a confusion matrix.
- Generate and display learning curves (training/validation accuracy and loss vs. epochs) to diagnose underfitting or overfitting.
- In Markdown cells, analyze your results:
 - Compare the performance of your baseline and improved models. Did the variation help? Why or why not?
 - Discuss the effects of data augmentation.
 - Analyze failure cases. Show some examples where your model failed and discuss potential reasons.

5. Conclusion

- Summarize your findings and the key insights from your experiments.
- Suggest potential directions for future work to further improve your model.

Notebook Structure and Content

- **Clarity is key.** Your notebook should be well-organized with clear headings (using Markdown) for each section.
- **Code cells** should be clean, well-commented, and executable from top to bottom without errors.
- **Markdown cells** should be used to provide context, explain your methodology, and interpret your results.
- **Outputs must be included.** Do not clear the outputs of your code cells before submitting. The grader should be able to see the printed metrics, tables, and plots you generated.
- **Academic Integrity:** Properly cite any external resources (datasets, code examples, papers) you referenced in your Markdown explanations.

Grading Criteria (Example Weights)

Criterion	Weight	Description
Problem Definition & Data Description	10%	Clarity of the problem statement; appropriate data selection, preparation, and explanation.
Model Implementation & Experiment Design	30%	Correct use of PyTorch, thoughtful architecture design, and a clear experimental plan (baseline vs. variant).
Experiment Execution & Results	25%	Quality of experiments; use of appropriate metrics; clear presentation of results within the notebook.
Analysis & Discussion	25%	Depth of insight in interpreting results; discussion of model performance, limitations, and failure cases.
Notebook Quality & Reproducibility	10%	Organization, writing clarity, and overall presentation; code is clean, commented, and runs successfully.
Creativity / Originality (Bonus)	0-5%	Innovative approaches, novel dataset usage, or thorough exploration beyond baseline requirements.

Use this guideline as a roadmap to structure your work. By systematically following these steps, you will develop a comprehensive and insightful mini-project.