Apertura: martes, 3 de febrero de 2026, 00:00
Cierre: martes, 10 de febrero de 2026, 00:00

Context and Motivation

In this course, neural networks are not treated as black boxes but as **architectural components** whose design choices affect performance, scalability, and interpretability.
This assignment focuses on **convolutional layers** as a concrete example of how inductive bias is introduced into learning systems.

Rather than following a recipe, students will **select, analyze, and experiment** with a convolutional architecture using a real dataset.

Learning Objectives

By completing this assignment, the student should be able to:

1. Understand the **role and mathematical intuition** behind convolutional layers.
2. Analyze how architectural decisions (kernel size, depth, stride, padding) affect learning.
3. Compare convolutional layers with fully connected layers for image-like data.
4. Perform a minimal but meaningful **exploratory data analysis (EDA)** for NN tasks.
5. Communicate architectural and experimental decisions clearly.

Dataset Selection (Student-Driven)

Each student must **choose one existing public dataset** suitable for convolutional neural networks.

Suggested sources (not mandatory)

- TensorFlow Datasets
- PyTorch `torchvision.datasets`
- Kaggle (only datasets that do not require competitions)

Dataset constraints

- Image-based (2D or 3D tensors)
- At least **2 classes**
- Dataset must fit in memory on a standard laptop or cloud notebook

Examples (illustrative, not restrictive):

- MNIST / Fashion-MNIST
- CIFAR-10 / CIFAR-100
- Medical images (X-ray, microscopy – small subsets)
- Satellite or land-use images

The student must **justify why the dataset is appropriate** for convolutional layers.

Assignment Tasks

1. Dataset Exploration (EDA)

Provide a concise analysis including:

- Dataset size and class distribution
- Image dimensions and channels
- Examples of samples per class
- Any preprocessing needed (normalization, resizing)

The goal is **understanding the structure**, not exhaustive statistics.

---

2. Baseline Model (Non-Convolutional)

Implement a **baseline neural network without convolutional layers**, e.g.:

- Flatten + Dense layers

Report:

- Architecture
- Number of parameters
- Training and validation performance
- Observed limitations

This establishes a reference point.

---

3. Convolutional Architecture Design

Design a CNN **from scratch**, not copied from a tutorial.

You must explicitly define and justify:

- Number of convolutional layers
- Kernel sizes
- Stride and padding choices
- Activation functions
- Pooling strategy (if any)

The architecture should be **simple but intentional**, not deep for its own sake.

---

4. Controlled Experiments on the Convolutional Layer

Choose **one aspect** of the convolutional layer and explore it systematically.

Examples (pick one):

- Kernel size (e.g. 3×3 vs 5×5)
- Number of filters
- Depth (1 vs 2 vs 3 conv layers)

- With vs without pooling
- Effect of stride on feature maps

Keep everything else fixed.

Report:

- Quantitative results (accuracy, loss)
- Qualitative observations
- Trade-offs (performance vs complexity)

---

5. Interpretation and Architectural Reasoning

Answer in your own words:

- Why did convolutional layers outperform (or not) the baseline?
- What inductive bias does convolution introduce?
- In what type of problems would convolution **not** be appropriate?

This section is graded heavily.

6. Deployment in Sagemaker

- Train the model in Sagemaker
- Deploy the model to a sagemaker endpoint

Deliverables

Git repository with.

1. **Notebook** (Jupyter):
    - Clean, executable, with explanations in Markdown
2. **Readme.md**:
    - Problem description
    - Dataset description
    - Architecture diagrams (simple)
    - Experimental results
    - Interpretation

Optional (bonus):

- Visualization of learned filters or feature maps

Evaluation Criteria (100 points)

- Dataset understanding and EDA: 15
- Baseline model and comparison: 15
- CNN architecture design and justification: 25
- Experimental rigor: 25

- Interpretation and clarity of reasoning: 20

Important Notes

- This is **not** a hyperparameter tuning exercise.
- Copy-paste architectures without justification will receive low scores.
- Code correctness matters less than **architectural reasoning**.