CSCI 145 Problem Set 7

October 8, 2025

Submission Instructions

Please upload *your* work by **11:59pm Monday October 13, 2025.**

- You are encouraged to discuss ideas and work with your classmates. However, you **must acknowledge** your collaborators at the top of each solution on which you collaborated with others and you **must write** your solutions independently.
- Your solutions to theory questions must be written legibly, or typeset in LaTeX or markdown. If you would like to use LaTeX, you can import the source of this document (available from the course webpage) to Overleaf.
- I recommend that you write your solutions to coding questions in a Jupyter notebook using Google Colab.
- You should submit your solutions as a **single PDF** via the assignment on Gradescope.

Grading: The point of the problem set is for *you* to learn. To this end, I hope to disincentivize the use of LLMs by **not** grading your work for correctness. Instead, you will grade your own work by comparing it to my solutions. This self-grade is due the Friday *after* the problem set is due, also on Gradescope.

Problem 1: Using and Interpreting CNNs

In this problem, we will explore convolutional neural networks (CNNs) and an interpetability tool called class activation maps (CAM). This tool from class may be helpful for visualizations.

Part A: Kernels by Hand

Write down one 3×3 kernel for blurring, one 3×3 kernel for detecting *horizontal* edges, and one 3×3 kernel for detecting *diagonal* edges. Confirm your kernels work as intended on example $2D$ matrices.

Part B: Pretrained CNN

Using `torch`, load a ResNet50 architecture (example code here). Preprocess an animal photo of your choice, pass it to the model, and inspect the predictions made by the model. In particular, print the five largest probabilities (apply softmax to the logits), and the corresponding human-readable labels.

Part C: Explaining Predictions via CAM

Setup. Let the last convolutional activation tensor be

$$\mathbf{A} \in \mathbb{R}^{B \times C \times H \times W},$$

where B is batch size, C the number of channels, and $H \times W$ the spatial resolution.

Apply *global adaptive average pooling* (GAP) over the spatial dimensions to obtain

$$\mathbf{z} = \mathrm{GAP}(\mathbf{A}) \in \mathbb{R}^{B \times C}, \qquad z_{b,c} = \frac{1}{HW} \sum_{x=1}^{H} \sum_{y=1}^{W} A_{b,c,x,y}.$$

The final fully connected (linear) classifier has weights

$$\mathbf{W} \in \mathbb{R}^{K \times C}, \quad \mathbf{b} \in \mathbb{R}^{K},$$

with K classes and row-vectors $\mathbf{w}_k^\top$ $(k = 1, \ldots, K)$. The class logits are

$$\mathbf{s} = \mathbf{z}\,\mathbf{W}^\top + \mathbf{b} \in \mathbb{R}^{B \times K}, \qquad s_{b,k} = \sum_{c=1}^{C} w_{k,c}\, z_{b,c} + b_k.$$

Class Activation Map (CAM). For class k, the CAM before pooling is defined (per sample b) as

$$M_{b,k}(x, y) = \sum_{c=1}^{C} w_{k,c}\, A_{b,c,x,y},$$

so that $\mathbf{M}_k \in \mathbb{R}^{B \times H \times W}$. For the *predicted* class of sample b,

$$k_b^* = \arg\max_k s_{b,k}, \qquad M_{b,k_b^*}(x, y) = \sum_{c=1}^{C} w_{k_b^*,c}\, A_{b,c,x,y}.$$

Procedure (what to implement).

1. Run a forward pass on the input image(s) and extract $\mathbf{A}$ (e.g., via a forward hook at the last conv layer).

2. Compute logits $\mathbf{s}$ and, for each sample b, find $k_b^* = \arg\max_k s_{b,k}$.

3. For each sample b, form the CAM using the corresponding classifier weights:

$$\mathbf{M}_b = \sum_{c=1}^{C} w_{k_b^*,c}\, \mathbf{A}_{b,c,\cdot,\cdot} \;\in\; \mathbb{R}^{H\times W}.$$

4. Upsample $\mathbf{M}_b$ to the input image resolution (H_0, W_0) using `torch.nn.functional.interpolate` (e.g., `mode='bilinear', align_corners=False`):

$$\tilde{\mathbf{M}}_b = \mathrm{Interp}(\mathbf{M}_b;\, \mathrm{size} = (H_0, W_0))\,.$$

5. Normalize $\tilde{\mathbf{M}}_b$ to $[0, 1]$ (e.g., min–max) and overlay it as a heatmap on the corresponding input image.

After overlaying the CAM on the input image, what regions appear most responsible for the model's prediction? What do you notice?

Problem 2: Transformers

Part A: Self-attention by Hand

You are given a sequence of $T = 3$ tokens with $d_{\text{model}} = 2$:

$$\mathbf{X} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}.$$

Let $d_k = d_v = 2$ and

$$\mathbf{W}_Q = \mathbf{W}_K = \mathbf{W}_V = \mathbf{I}_2.$$

1. Compute $\mathbf{Q} = \mathbf{X}\mathbf{W}_Q,\ \mathbf{K} = \mathbf{X}\mathbf{W}_K,\ \mathbf{V} = \mathbf{X}\mathbf{W}_V$.
2. Form the unnormalized score matrix $\mathbf{S} = \mathbf{Q}\mathbf{K}^\top/\sqrt{2}$.
3. Apply a *causal mask* (token t may only attend to $\{1, \ldots, t\}$; set masked entries to $-\infty$).
4. Apply `softmax` row-wise to obtain attention weights $\mathbf{A}$.
5. Compute outputs $\mathbf{Y} = \mathbf{A}\mathbf{V}$.

Report $\mathbf{S}$ (masked), $\mathbf{A}$, and $\mathbf{Y}$.

Part B: Self-attention Activations

Consider a pretrained transformer model. Select a text input of your choice. Plot the self-attention weights at several layers (both early and later) in your transformer model. What patterns do you notice?

Part C: Cross-attention Activations

Consider a pretrained transformer model for translation. Select a text input of your choice and translate it. Now feed the original and translated text to a translation model and plot the cross-attention weights at several layers (both early and later). What patterns do you notice?