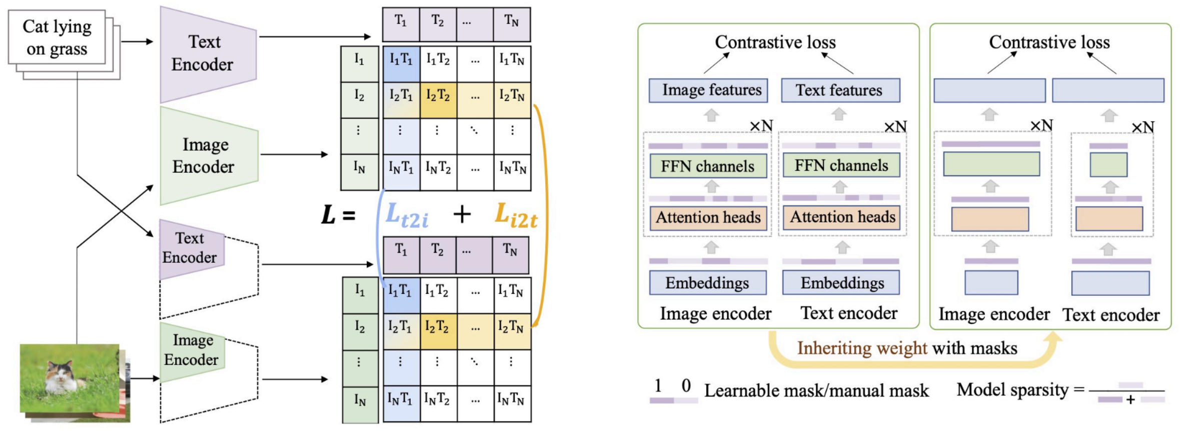
LAB 1: Adversarial Attacks on VLMs
Due: 10/23/2025

Students will implement an adversarial perturbation attack against a vision–language model that accepts an image + text prompt and returns a text response. Attacks may modify **image pixels**, **input text tokens**, or **both**. Teams will be evaluated by a combined cost/score that accounts for (1) number of pixels changed, (2) number of tokens changed, (3) attack success rate, and (4) average number of model queries required.

Labs must be done in teams of 2.

Background — What are Vision–Language Models?

Vision–language models (VLMs) are multimodal neural networks that jointly process visual inputs (images) and textual inputs (prompts, questions) and produce textual outputs (captions, answers, instructions). They typically combine a convolutional or transformer-based image encoder with a transformer-based language model, bridging visual features to natural language via cross-modal attention or learned projection layers. VLMs power tasks such as image captioning, visual question answering (VQA), multimodal instruction following, and more. Because they jointly depend on both image and text inputs, adversarial attacks can target either modality (images or text) or both together, creating novel attack surfaces.



Affinity mimicking

Weight inheritance

Task Description (detailed)

- **Attack goal.** Given a stacked input `(image, prompt)` for [TinyCLIP](), craft minimal perturbations to the image pixels and/or to the input tokens such that the model's output meets a specified *success condition* (e.g., outputs a target phrase, changes classification to a target label, or fails to answer correctly).

- **Allowed modifications.**

    - **Image perturbations:** Any pixel-level modification to the input image is allowed, subject to these constraints:

        - Result must be in the valid pixel range (e.g., 0–255).

        - You may apply dense perturbations (many pixels) or sparse (few pixels). The scoring metric will penalize the number of pixels changed.

        - You may use gradient-based or gradient-free methods. White box access to the model is provided.

    - **Text perturbations:** You may change tokens in the textual prompt (substitutions only) that are actually presented to the model. Constraints:

        - The text after modification must be valid UTF-8 and not violate any length limits. [ONLY VALID TOKEN SUBSTITUTIONS ALLOWED]
        - Each altered token counts toward the token-modification cost. Levenshtein distance is used to measure the caption similarity.

    - **Combined attacks:** You may change both image and text simultaneously. The scoring will reflect both kinds of edits.

- **Queries / cost of interaction.**

    - Each time you submit an input pair `(image, prompt)` to `TinyCLIP` to obtain a model output counts as **1 query**.

    - Your evaluation will measure the **average number of queries** used per successful attack.

- **Success condition.**
    - If the model decision is flipped under the given budgets, the attack is considered successful.

Evaluation & Scoring

Important: The instructor will paste the exact cost function here. Implement your system to compute a score using that cost function. Below is the information you must provide outputs for; the instructor's function will combine these into a single scalar score.

You must report the following for each attack run and aggregated across the dataset:

1. **Attack success rate (ASR)** — fraction (%) of target examples where the attack met the success condition.

2. **Average pixels modified (APM)** — average (over successful attacks, or as defined) number of distinct pixels whose values changed relative to the original image. Full cost for the first pixel, half cost for the rest of the continuous pixels.

3. **Average tokens modified (ATM)** — average number of input tokens changed (measured using the model tokenizer).

4. **Average queries (AQ)** — average number of model queries required per successful attack (or per attempt if instructor requests).

Attack will be scored as:

```
ASR - 0.5×(APM + 10*ATM)/(10*T_MAX + P_MAX) - 0.1×(AQ/Q_MAX)
```

What to Submit

We will be providing a Python notebook with the VLM model and helper functions. Your goal is to update the notebook with your attack which will be your main submission. We will be running your submitted notebooks on the sample inputs provided as well as unseen test inputs to score your solutions.

In addition, please submit a 5-slide deck detailing your proposed method and results. We will send you the format for the slide deck shortly. All materials for this lab (code, data, files) will be available on Brightspace.

Scoring:

- 25 Points: reproducibility on sample inputs provided.
- 25 Points: results within tolerable range on hidden inputs (drawn from the same distribution)
- 25 Points: competitively awarded based on team rank
- 25 Points: main ideas, rigor and/or novelty of methods as inferred from slide-deck and submitted code.