Deep Learning Term Project CS60010 Spring Semester 2024

Task: Build encoder decoder models for Automatic Image Captioning

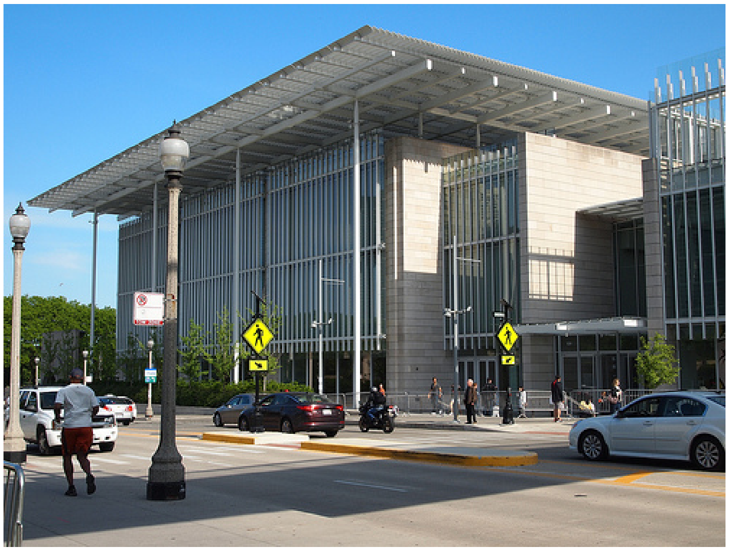
Deadline: 14th April, 2024 EOD

Dataset:
https://drive.google.com/file/d/1FMVcFM78XZE1KE1rlkGBpCdcdI58S1LB/view?usp=sharing

Related papers: https://aclanthology.org/P18-1238.pdf
https://arxiv.org/pdf/1411.4555.pdf

Description:
Given an input image, generate a caption for the image.

Input:



Output: A large building with bars on the windows in front of it. There are people walking in front of the building. There is a street in front of the building with many cars on it.

Evaluation metrics:
CIDEr
ROUGE-L
SPICE

Subtasks:

Part A

Design a simple CNN-based encoder for the image and RNN-based decoder model for the task and report results on the given test set [Refer to the Related Papers and Resources section for help].

Part B

Design a transformer based encoder decoder model using a Vision Transformer (ViT) as the image encoder (You can try using a lightweight vit model like vit-small-patch16-224) and a text decoder of your choice. You can get creative and use any models that fit into the Google Colab or Kaggle GPU. To ensure fair evaluation, we will not accept models which take up more than 15GB of GPU space (which is the limit on the T4 GPU available on the Google Colab Free Tier).

Deliverables:

- Python notebooks for Part A and Part B. Make sure the last cell of the notebook contains the generated captions of the test set.
- README file containing running instructions (or any other additional info). Also include names and roll numbers of ALL team members.
- Project Report containing the following sections (max pages: 4):
    - Methodology (with detailed explanation and diagram for each model)
    - Results (with table containing eval metrics for test set) and Analysis

General Guidelines:

- You can use Colab or Kaggle for GPU.
- You cannot use end-to-end trained models available on huggingface or github. Your notebooks must clearly contain the model class, showing the design of the model.
- Pre trained encoders or decoders may be used for Part B, but they must be fine tuned on the train set.
- You may not use any train set other than the one given for finetuning/training your models.
- Please follow the following program format:
    - Preprocessing (Data parsing and formatting)
    - Model Creation
    - Model Training (using train set) and validation after each epoch (on the val set).
    - Model Evaluation on the test set.
- You are allowed to use either pytorch or tensorflow.
- Before each section of the code in the notebook, write a brief description in a text cell.
- You may use libraries/boilerplate code for calculating the evaluation metrics.
- You are compulsorily required to show the output of your model on the test set, as output in the ipython notebook.
- You may use https://app.diagrams.net/ to make the diagrams for your report.
- Remember to fine tune your hyperparameters!

Submission Guidelines:

- All the deliverables are to be submitted in a zip file, with the naming format team_id_<num>*_project.zip. (Eg: team*_id_1_project.zip for Team 1)

- The notebooks are to be named team_id_<num>_a.ipynb and team_id_<num>_b.ipynb respectively (Eg: team_id_1_a.ipynb, team_id_1_b.ipynb for Team 1).
- Report should be named team_id_<num>_report.pdf.
- There should be one submission per team.

IMPORTANT: PLEASE FOLLOW THE NAMING CONVENTIONS STRICTLY. FAILURE TO DO SO WILL RESULT IN DEDUCTION OF MARKS.

Plagiarism: We will be employing strict plagiarism checking. If your code matches with another team's code, all those students will be awarded zero marks for the assignment. Therefore, please ensure there is no sharing of code. Models may be coincidentally the same. **Unfair usage of LLMs (like ChatGPT, Gemini, Claude, etc) to complete the assignment will also be heavily penalized.**

Code Error: If the code doesn't run for a particular experiment, partial marks may be awarded based on structure and logic of code. If required, you may be contacted by the TAs to explain your code.

Resources:
Encoder Decoder Models https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning
https://huggingface.co/docs/transformers/en/model_doc/vision-encoder-decoder.
RNN coding tutorial
https://www.kaggle.com/code/kanncaa1/recurrent-neural-network-with-pytorch \
Image Captioning Coding Tutorial
https://www.youtube.com/watch?v=y2BaTt1fxJU&list=PLCJHEFznK8ZybO3cpfWf4gKbyS5VZgppW&index=1

Tentative Marking Scheme (subject to change): [Total: 45]

Part A [15 marks] - containing proper sections and explanations.
- Preprocessing (Data parsing and formatting) [2.5]
- Model Creation [7.5]
- Model Training and Validation (using train and val set) [2.5]
- Model Evaluation on the Test set [2.5]

Part B [20 marks] - containing proper sections and explanations.
- Preprocessing (Data parsing and formatting) [2.5]
- Model Creation [12.5]
- Model Training and Validation (using train and val set) [2.5]
- Model Evaluation on the Test set [2.5]

Report [10 marks]