HOMEWORK 3

16824 VISUAL LEARNING AND RECOGNITION (SPRING 2024)
https://piazza.com/class/lr8dzk3rn9n4d8

RELEASED: Sat, 23rd Mar 2024
DUE: Wed, 10th Apr 2024
Instructor: Deepak Pathak
TAs: Mihir Prabhudesai, Kenny Shaw, Shagun Uppal, Himangi Mittal, Sayan Mondal

START HERE: Instructions

- **Collaboration policy:** All are encouraged to work together BUT you must do your own work (code and write up). If you work with someone, please include their name in your write-up and cite any code that has been discussed. If we find highly identical write-ups or code or lack of proper accreditation of collaborators, we will take action according to strict university policies. See the Academic Integrity Section detailed in the initial lecture for more information.
- **Late Submission Policy:** There are a **total of 10** late days across all homework submissions. Submissions that use additional late days will incur a 10% penalty per late day.
- **Submitting your work:**
  - We will be using Gradescope (`https://gradescope.com/`) to submit the Problem Sets. Please use the provided template only. Submissions must be written in LaTeX. All submissions not adhering to the template will not be graded and receive a zero.
  - **Deliverables:** Please submit all the `.py` files. Add all relevant plots and text answers in the boxes provided in this file. TO include plots you can simply modify the already provided latex code. Submit the compiled `.pdf` report as well.

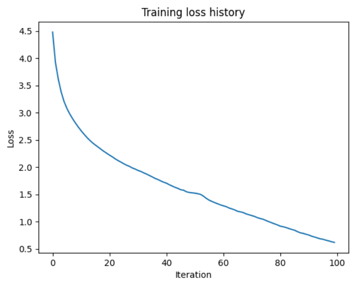
NOTE: Partial points will be given for implementing parts of the homework even if you don't get the mentioned numbers as long as you include partial results in this pdf.

1 Image Captioning with Transformers (70 points)

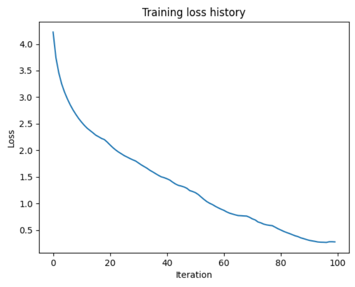
We will be implementing the different pieces of a Transformer decoder (Transformers), and train it for image captioning on a subset of the COCO dataset.

- **Setup:** Run the following command to extract COCO data, in the `transformer_captioning/datasets` folder : `./get_coco_captioning.sh`
- **Question:** Follow the instructions in the `README.md` file in the `transformer_captioning` folder to complete the implementation of the transformer decoder.
- **Deliverables:** After implementing all parts, use run.py for training the full model. The code will log plots to `plots`. Extract plots and paste them into the appropriate section below.
- **Expected results:** These are expected training losses after 100 epochs. Do not change the seed in run.py.
  - 2-heads, 2-layers, lr 1e-4: Final loss ≤ 1
  - 4-heads, 6-layers, lr 1e-4: Final loss ≤ 0.3
  - 4-heads, 6-layers, lr 1e-3: Final loss ≤ 0.05

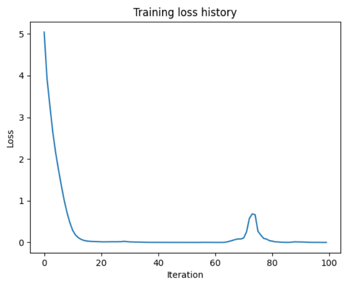
1. Paste training loss plots for each of the three hyper-param configs
   2-heads-2-layers-lr-1e-4: **TODO: fill in final train loss here.**
   4-heads-6-layers-lr-1e-4: **TODO: fill in final train loss here.**
   4-heads-6-layers-lr-1e-3: **TODO: fill in final train loss here.**



(a) 2-heads-2-layers-lre-4



(b) 4-heads-6-layers-lre-4

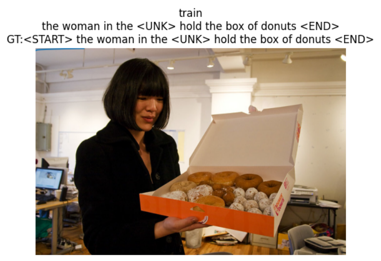


(c) 4-heads-6-layers-lre-3

2. Paste any three generated captioning samples from the training set with the three different settings. The provided code creates these plots at the end of training.

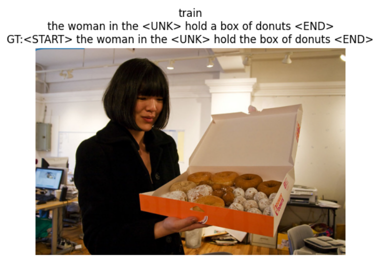
train
the man in the kitchen with the <UNK> the the woman <UNK> <END>
GT:<START> the woman in the <UNK> hold the box of donuts <END>

(a) Sample1: 2-heads-2-layers-lre-4



train
the woman in the <UNK> hold the box of donuts <END>
GT:<START> the woman in the <UNK> hold the box of donuts <END>

(b) Sample2:4-heads-6-layers-lre-4



train
the woman in the <UNK> hold a box of donuts <END>
GT:<START> the woman in the <UNK> hold the box of donuts <END>

(c) Sample3:4-heads-6-layers-lre-3

3. Based on the observations of the three different settings, What would you change in the training procedure to get better validation performance? Why tweaking these hyper-parameters will lead to better performances?

Solution: Given the performance outcomes of Transformer models under three varied setting configurations for image captioning on the COCO dataset, adjustments to three critical hyperparameters (number of attention heads, layers, and learning rate) provided insightful intuition and observations for optimizing validation set accuracy.

Hyperparameter Tuning Strategy:

1. *Number of Attention Heads and Layers*: Empirical evidence from observations from our experiments suggests that models with higher complexity exhibit improved performance, attributed to an improved capacity to process and represent data. Incrementally increasing the number of heads and layers could further refine the model's ability to generalize, crucial for validation accuracy. This is because more attention heads enable diversified parallel processing of data representations, and additional layers allow for learning complex hierarchies and structures in data. Going from the model complexity of number of attention heads of 2 and number of layers of 2 to a model complexity of number of attention heads of 4 and number of layers of 6, while keeping the learning rate constant at 1e-4 for both cases, a marked improvement in the captions of the validation set was observed (albeit still quite inaccurate) as it started to pick up on individual items or entities in the validation set images.

2. *Learning Rate*: The effectiveness of a higher learning rate in complex models indicates the necessity for an aggressive yet balanced optimization strategy. A learning rate scheduler that adjusts based on validation loss can dynamically fine-tune this balance, improving the model's exploration of the parameter space and, consequently, validation performance. Going from a learning rate of 1e-4 to 1e-3 while keeping the model complexity of number of attention heads of 4 and number of layers of 6, a marked improvement in the captions of the validation set was observed (albeit still quite inaccurate) as it started to pick up on individual items or entities in the validation set images.

Intuition Behind Observations: The interplay between model complexity and learning rate adjustments reveals a balance essential for effective learning and generalization. Increased model complexity offers deeper, more nuanced data processing capabilities, while optimal learning rate adjustments ensure efficient parameter space navigation without the pitfall of instability. From the observation, the configuration with 4 attention heads, 6 layers, and a learning rate of 1e-3 produced the closest captions to the ground truth for the validation set.

Additional Strategies for Validation Performance Improvement:

- Integrate dropout in strategic locations within the Transformer architecture to mitigate overfitting. Additionally, experiment with weight decay via optimizers like AdamW to control model complexity, improving the model's generalization capabilities on unseen data.

- Apply learning rate schedulers such as the Cosine Annealing or Exponential Decay to adapt the learning rate during training dynamically. This approach helps in navigating the optimization landscape more effectively, potentially leading to improved validation accuracy.

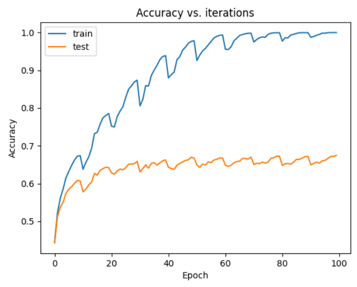
- Leverage pre-trained models for both the image feature extraction and the initialization of the transformer's embedding layers. This can provide a strong prior knowledge base, accelerating the learning process and enhancing model performance on validation data.

- Implement a curriculum learning strategy, gradually increasing the complexity of training samples. This method helps the model learn basic patterns before advancing to more complex relationships, potentially improving its generalization to the validation set.

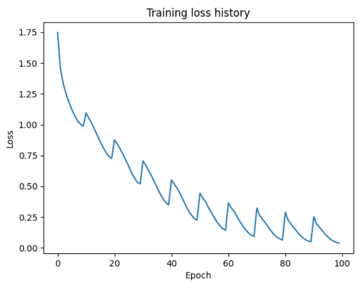
2 Classification with Vision Transformers (30 points)

We will use the transformer you implemented in the previous part to implement a Vision Transformer (ViT), for classification on CIFAR10.

- **Question:** Follow the instructions in the `README.md` file in the `vit_classification` folder. You are encouraged to resuse code from the previous question.
- **Deliverables:** Run training using `run.py` for training the full model. The code will log plots `acc_out.png` (train and test accuracy) and `loss_out.png` (train loss).
- **Expected Results:** After 100 epochs, test accuracy should be $\geq 65\%$, train accuracy should be $\approx 100\%$, and training loss ≤ 0.3.



(a) Train/test accuracy



(b) Training loss

Collaboration Survey Please answer the following:

1. Did you receive any help whatsoever from anyone in solving this assignment?

   [✓ Yes] No

   - If you answered 'Yes', give full details:
   - (e.g. "Jane Doe explained to me what is asked in Question 3.4")

   Yes, I did use a few sources on articles on the internet, Stack Overflow, GitHub, and ChatGPT to help with the assignment (particularly troubleshooting errors). They are referenced as follows:

   (a) https://medium.com/@seffa.b/unlocking-visual-information-a-guide-to-image-captioning-with-transformers-9693ced8acc3
   (b) https://medium.com/@mayankkeshari34/image-captioning-using-transformers-627cd39c6b7a
   (c) https://github.com/saahiluppal/catr
   (d) https://github.com/senadkurtisi/pytorch-image-captioning
   (e) https://medium.com/@curttigges/building-the-vision-transformer-from-scratch-d77881edb5ff
   (f) https://medium.com/machine-intelligence-and-deep-learning-lab/vit-vision-transformer-cc56c8071a20
   (g) https://medium.com/@hansahettiarachchi/unveiling-vision-transformers-revolutionizing-computer-vision-beyond-convolution-c410110ef061
   (h) https://github.com/lucidrains/vit-pytorch

2. Did you give any help whatsoever to anyone in solving this assignment?

   Yes [✓ No]

   - If you answered 'Yes', give full details:
   - (e.g. "I pointed Joe Smith to section 2.3 since he didn't know how to proceed with Question 2")

3. Note that copying code or writeup even from a collaborator or anywhere on the internet violates the Academic Integrity Code of Conduct.