Reading (Noisy) Captions Embedded in Images

Total points: 100
Due Date: Friday Nov 26, 2021. 11:50 pm

Notes

- This assignment has two parts - Non Competitive and Competitive.
- You should submit all your code (including any pre-processing scripts written by you) and any graphs that you might plot.
- Do not submit the datasets. Do not submit any code that we have provided to you for processing.
- Include **a write-up (pdf) file**, which includes a brief description for each question explaining what you did. Include any observations and/or plots required by the question in this single write-up file.
- **You should use Python as your programming language and PyTorch as the deep learning framework.**
- **Your code should have appropriate documentation for readability.**
- No buffer days are allowed for this assignment.
- You will be graded based on what you have submitted as well as your ability to explain your code.
- This assignment is supposed to be done in **teams of 2**. You should carry out all the implementation by yourself.
- We plan to run Moss on the submissions. We will **also include submissions from the internet** to maintain integrity. Any cheating will result in a zero on the assignment and possibly much stricter penalties (including a **fail grade** and/or a **DISCO**).

1 Dataset Links

1. Train images: link
2. Train captions: link
3. Test images: link

2 Non-Competitive Part (40 points)

You are given a dataset of images in which English captions are embedded in the image itself. The images are named as image_id.jpg where id is the image index. **Unfortunately, these captions may or may not correctly correspond to the image in which it is embedded i.e. in some cases it has been randomly embedded on top of the background image and in the remaining examples it's correctly embedded on top of correct image. The task is to predict the text embedded into the image (irrespective of the background image itself)**. To make it a bit challenging, the text might be rotated , translated and it's font (also font size) might be different for different images. The Train_text.tsv file contain the captions of the images, with each line (tab separated) format having image id and the corresponding captions. You will use the Encoder-Decoder architecture for modelling this problem. An encoder is used to encode the input into a vector representation and a decoder is applied on this vector representation to generate the sequence auto-regressively (one word/character at a time). You have to implement the encoder for the images and a decoder for text captions in this part along with beam search to find the most optimal sequence. You can use the following as a starting point -

1. **Encoder:** Design a CNN based encoder that handles the variable sized images.
2. **Decoder:** Design a RNN / LSTM based decoder which generates the captions given the encoded image input. Note that you can either design a word-level or a character-level LSTM for generating the caption.
3. **Training strategy:** Use cross-entropy as the loss function and **teacher-forcing** for training the decoder. Don't forget to use START and END tokens to allow variable length caption outputs in the decoder.
4. **Inference at test time**: Instead of using the token with maximum probability at each step (Greedy Decoding) for generating the tokens (words) in the caption, you will use Beam Search for generating your captions. It is a Dynamic Programming method to get better sequences than simple Greedy Decoding. Read Section 4 of **this pdf document**. We also recommended you to read Sections 1.2, 1.3, and 1.4 for better understanding.

5. You have to generate the captions (verbatim as embedded in image) for each image using Beam Search as described above. We will use BLEU (BiLingual Evaluation Understudy) scores for evaluating your model performance. You can read more about it in Section 5.3 of **this pdf document**.

3 Competitive Part (60 points)

You are allowed to use any pre-trained CNN models for your encoder and pre-trained word vectors for your decoder. Any generic pre-trained models are allowed. **But, you are not allowed to use a complete end-to-end trained model for image captioning or related tasks from the internet. In case of any doubt regarding what existing code is allowed, you should contact us.**

4 Resources

You can refer to this paper to get an in-depth understanding of Image Captioning Encoder-Decoder framework. Though our task is not exactly about image captioning, this literature may still be useful for you to get started. We also provide the boiler plate code with base classes to make it easier to implement the assignment. You can download it from this link. This is an ipynb file that can be opened using jupyter notebook or jupyter lab. Note that this is just for beginners and you may suitably modify it for image/text pre-processing on your own.

5 Submission Instructions

On Moodle, submit a single zip file named **<team_name>.zip** where **<team_name>** needs to be filled in this google form by Nov 3rd. Note that any whitespaces are not allowed in the team name (underscores are allowed, e.g. dungeon_master is acceptable).
Note:- Unzipping this zip must generate a folder by the same name in current working directory. This folder must contain a .tsv file by the name **<team_name>.csv** along with your code and report. **The <team_name>.csv should have exactly the same format as Train_text.tsv i.e. image id followed by tab followed by predicted captions in every line. Only one of you** should make the submission.
Sample submission file -

<image_id_1><\tab><caption_1>
......
......
<image_id_n><\tab><caption_n>

A sample submission file can be downloaded from this link. **Very important:- We will download your submissions from moodle and run auto-evaluation script to get leaderboard scores. So it's extremely important to strictly adhere to these guidelines and directory structure, otherwise you will be penalised heavily and your submission won't be considered for leaderboard (0 marks will be given for competitive part).**