2_Training

October 19, 2024

1 Computer Vision Nanodegree

1.1 Project: Image Captioning

In this notebook, you will train your CNN-RNN model.

You are welcome and encouraged to try out many different architectures and hyperparameters when searching for a good model.

This does have the potential to make the project quite messy! Before submitting your project, make sure that you clean up: - the code you write in this notebook. The notebook should describe how to train a single CNN-RNN architecture, corresponding to your final choice of hyperparameters. You should structure the notebook so that the reviewer can replicate your results by running the code in this notebook.
- the output of the code cell in **Step 2**. The output should show the output obtained when training the model from scratch.

This notebook **will be graded**.

Feel free to use the links below to navigate the notebook: - Step 1: Training Setup - Step 2: Train your Model - Step 3: (Optional) Validate your Model

Step 1: Training Setup

In this step of the notebook, you will customize the training of your CNN-RNN model by specifying hyperparameters and setting other options that are important to the training procedure. The values you set now will be used when training your model in **Step 2** below.

You should only amend blocks of code that are preceded by a `TODO` statement. **Any code blocks that are not preceded by a `TODO` statement should not be modified**.

1.1.1 Task #1

Begin by setting the following variables: - `batch_size` - the batch size of each training batch. It is the number of image-caption pairs used to amend the model weights in each training step. - `vocab_threshold` - the minimum word count threshold. Note that a larger threshold will result in a smaller vocabulary, whereas a smaller threshold will include rarer words and result in a larger vocabulary.
- `vocab_from_file` - a Boolean that decides whether to load the vocabulary from file. - `embed_size` - the dimensionality of the image and word embeddings.
- `hidden_size` - the number of features in the hidden state of the RNN decoder.

- `num_epochs` - the number of epochs to train the model. We recommend that you set `num_epochs=3`, but feel free to increase or decrease this number as you wish. This paper trained a captioning model on a single state-of-the-art GPU for 3 days, but you'll soon see that you can get reasonable results in a matter of a few hours! (*But of course, if you want your model to compete with current research, you will have to train for much longer.*) - `save_every` - determines how often to save the model weights. We recommend that you set `save_every=1`, to save the model weights after each epoch. This way, after the `i`th epoch, the encoder and decoder weights will be saved in the `models/` folder as `encoder-i.pkl` and `decoder-i.pkl`, respectively. - `print_every` - determines how often to print the batch loss to the Jupyter notebook while training. Note that you **will not** observe a monotonic decrease in the loss function while training - this is perfectly fine and completely expected! You are encouraged to keep this at its default value of `100` to avoid clogging the notebook, but feel free to change it. - `log_file` - the name of the text file containing - for every step - how the loss and perplexity evolved during training.

If you're not sure where to begin to set some of the values above, you can peruse this paper and this paper for useful guidance! **To avoid spending too long on this notebook**, you are encouraged to consult these suggested research papers to obtain a strong initial guess for which hyperparameters are likely to work best. Then, train a single model, and proceed to the next notebook (**3_Inference.ipynb**). If you are unhappy with your performance, you can return to this notebook to tweak the hyperparameters (and/or the architecture in **model.py**) and re-train your model.

Some of the

1.1.2 Question 1

Question: Describe your CNN-RNN architecture in detail. With this architecture in mind, how did you select the values of the variables in Task 1? If you consulted a research paper detailing a successful implementation of an image captioning model, please provide the reference.

Answer:

I consulted the two papers listed above "Show and Tell: A Neural Image Caption Generator" and "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention" to gain a better understanding on the overall process and framework to implement an Image Caption.

The overall approach includes an Encoder and a Decoder. The **Encoder** leveraged a pre-trained CNN architecture like `ResNet-50` to help extract features from the images. The line `modules = list(resnet.children())[:-1]` is when we remove the last Fully Connected layer in the pre-trained architecture. The goal of the encoder is to enode the content of the image into a smaller feature vector. Which later gets passed to the Decoder (RNN) network.

The **Decoder** which is our LSTM that we use to generate the captions for an image. We pass the encoder embedding as input to the LSTM to learn. The vocabulary in the trainnig pool are pretty much all the unique words in our data set, with the added and tokens to indicate start and end of a sentence.

I used a `vecob_threshold` of 4. The number of words were better than 5, and I did not want to increase the vocab words to include every rare word that may have only been used once. A balance would be key, so I went with 4. If I would test and rerun the training again, I would like to try 2-3 to evaluate the generated captions and the quality of the output.

for the `batch_size` I think 10 was too small. If I had the bandwitdh to rerun this again, I would probably increase this value and the size of each batch to maybe 32.

`embed_size` is used both in embedded image feature vector and word embedding. In this case I used 256 and a `hidden_size` of 512.

1.1.3 (Optional) Task #2

Note that we have provided a recommended image transform `transform_train` for pre-processing the training images, but you are welcome (and encouraged!) to modify it as you wish. When modifying this transform, keep in mind that: - the images in the dataset have varying heights and widths, and - if using a pre-trained model, you must perform the corresponding appropriate normalization.

1.1.4 Question 2

Question: How did you select the transform in `transform_train`? If you left the transform at its provided value, why do you think that it is a good choice for your CNN architecture?

Answer: The `transform_train` was not changed. It covered resizing, random cropping, random horizontal flip, converting to tensor, followed by normalization. This is a standard image transformer that I felt was more than sufficient for the current task. These transformations are what is needed for the `ResNet50` since it is a pre-trained model we need to make sure the images are in the expected size and normal form.

1.1.5 Task #3

Next, you will specify a Python list containing the learnable parameters of the model. For instance, if you decide to make all weights in the decoder trainable, but only want to train the weights in the embedding layer of the encoder, then you should set `params` to something like:

```
params = list(decoder.parameters()) + list(encoder.embed.parameters())
```

1.1.6 Question 3

Question: How did you select the trainable parameters of your architecture? Why do you think this is a good choice?

Answer: For the trainable parameters I made all the weights int he decoder trainable, which at every iteration they get updated. For the encoder, I used to only updated the weights in the emedding layer.

1.1.7 Task #4

Finally, you will select an optimizer.

1.1.8 Question 4

Question: How did you select the optimizer used to train your model?

Answer: I selected Adam optimizer, instead of SGD. From past experiences I have seen it perform/converge faster. When I first started the training I used the wrong learning rate of 0.01 which

was too high and there was no improvement and had to end it after 1 epoch. I updated the Learning Rate to 0.001 and noticed signifcant improvement in the overall learning.

[11]:
```
import torch
import torch.nn as nn
from torchvision import transforms
import sys
sys.path.append('/opt/cocoapi/PythonAPI')
from pycocotools.coco import COCO
from data_loader import get_loader
from model import EncoderCNN, DecoderRNN
import math


## Select appropriate values for the Python variables below.
batch_size = 10          # batch size
vocab_threshold = 4        # minimum word count threshold
vocab_from_file = True      # if True, load existing vocab file
embed_size = 256           # dimensionality of image and word embeddings
hidden_size = 512          # number of features in hidden state of the RNN␣
 ↪decoder
num_epochs = 1  #3             # number of training epochs
save_every = 1             # determines frequency of saving model weights
print_every = 100          # determines window for printing average loss
log_file = 'training_log.txt'       # name of file with saved training loss and␣
 ↪perplexity

# (Optional)  Amend the image transform below.
transform_train = transforms.Compose([
    transforms.Resize(256),                          # smaller edge of image␣
 ↪resized to 256
    transforms.RandomCrop(224),                      # get 224x224 crop from␣
 ↪random location
    transforms.RandomHorizontalFlip(),               # horizontally flip image␣
 ↪with probability=0.5
    transforms.ToTensor(),                           # convert the PIL Image to␣
 ↪a tensor
    transforms.Normalize((0.485, 0.456, 0.406),      # normalize image for␣
 ↪pre-trained model
                         (0.229, 0.224, 0.225))])

# Build data loader.
data_loader = get_loader(transform=transform_train,
                         mode='train',
                         batch_size=batch_size,
                         vocab_threshold=vocab_threshold,
                         vocab_from_file=vocab_from_file)
```

```
# The size of the vocabulary.
vocab_size = len(data_loader.dataset.vocab)

# Initialize the encoder and decoder.
encoder = EncoderCNN(embed_size)
decoder = DecoderRNN(embed_size, hidden_size, vocab_size)

# Move models to GPU if CUDA is available.
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
encoder.to(device)
decoder.to(device)

# Define the loss function.
criterion = nn.CrossEntropyLoss().cuda() if torch.cuda.is_available() else nn.
 ↪CrossEntropyLoss()

# Specify the learnable parameters of the model.
params = list(decoder.parameters())+list(encoder.embed.parameters())

# Define the optimizer.
optimizer = torch.optim.Adam(params,lr=0.001)

# Set the total number of training steps per epoch.
total_step = math.ceil(len(data_loader.dataset.caption_lengths) / data_loader.
 ↪batch_sampler.batch_size)
```

```
Vocabulary successfully loaded from vocab.pkl file!
loading annotations into memory…
Done (t=0.43s)
creating index…
index created!
Obtaining caption lengths…


  0%|
| 0/414113 [00:00<?, ?it/s]
  0%|
| 1653/414113 [00:00<00:24, 16528.68it/s]
  1%|
| 3402/414113 [00:00<00:24, 17093.77it/s]
  1%|
| 5200/414113 [00:00<00:23, 17492.82it/s]
  2%|
| 6974/414113 [00:00<00:23, 17589.18it/s]
  2%|
| 8736/414113 [00:00<00:23, 17599.86it/s]
  3%|
```

```
| 10503/414113 [00:00<00:22, 17621.70it/s]
  3%|
| 12266/414113 [00:00<00:22, 17617.19it/s]
  3%|
| 14068/414113 [00:00<00:22, 17744.23it/s]
  4%|
| 15858/414113 [00:00<00:22, 17792.05it/s]
  4%|
| 17638/414113 [00:01<00:22, 17764.22it/s]
  5%|
| 19441/414113 [00:01<00:22, 17844.89it/s]
  5%|
| 21245/414113 [00:01<00:21, 17903.96it/s]
  6%|
| 23071/414113 [00:01<00:21, 18010.91it/s]
  6%|
| 24873/414113 [00:01<00:21, 18012.65it/s]
  6%|
| 26675/414113 [00:01<00:21, 17997.00it/s]
  7%|
| 28485/414113 [00:01<00:21, 18026.03it/s]
  7%|
| 30288/414113 [00:01<00:21, 18009.99it/s]
  8%|
| 32107/414113 [00:01<00:21, 18063.30it/s]
  8%|
| 33914/414113 [00:01<00:21, 17929.71it/s]
  9%|
| 35708/414113 [00:02<00:21, 17905.25it/s]
  9%|
| 37499/414113 [00:02<00:21, 17873.53it/s]
  9%|
| 39290/414113 [00:02<00:20, 17881.83it/s]
 10%|
| 41079/414113 [00:02<00:20, 17854.74it/s]
 10%|
| 42865/414113 [00:02<00:20, 17751.91it/s]
 11%|
| 44641/414113 [00:02<00:20, 17736.10it/s]
 11%|
| 46439/414113 [00:02<00:20, 17807.90it/s]
 12%|
| 48222/414113 [00:02<00:20, 17811.98it/s]
 12%|
| 50004/414113 [00:02<00:20, 17760.28it/s]
 13%|
| 51795/414113 [00:02<00:20, 17802.34it/s]
 13%|
```

```
| 53590/414113 [00:03<00:20, 17845.50it/s]
 13%|
| 55398/414113 [00:03<00:20, 17914.53it/s]
 14%|
| 57194/414113 [00:03<00:19, 17925.86it/s]
 14%|
| 58994/414113 [00:03<00:19, 17947.38it/s]
 15%|
| 60795/414113 [00:03<00:19, 17965.13it/s]
 15%|
| 62596/414113 [00:03<00:19, 17977.55it/s]
 16%|
| 64412/414113 [00:03<00:19, 18030.80it/s]
 16%|
| 66227/414113 [00:03<00:19, 18064.63it/s]
 16%|
| 68034/414113 [00:03<00:19, 18019.27it/s]
 17%|
| 69836/414113 [00:03<00:19, 18003.67it/s]
 17%|
| 71655/414113 [00:04<00:18, 18057.45it/s]
 18%|
| 73461/414113 [00:04<00:18, 18049.02it/s]
 18%|
| 75266/414113 [00:04<00:18, 17966.91it/s]
 19%|
| 77063/414113 [00:04<00:18, 17950.03it/s]
 19%|
| 78877/414113 [00:04<00:18, 18005.26it/s]
 19%|
| 80698/414113 [00:04<00:18, 18065.47it/s]
 20%|
| 82506/414113 [00:04<00:18, 18067.29it/s]
 20%|
| 84313/414113 [00:04<00:18, 18040.02it/s]
 21%|
| 86118/414113 [00:04<00:18, 17991.71it/s]
 21%|
| 87933/414113 [00:04<00:18, 18035.98it/s]
 22%|
| 89737/414113 [00:05<00:17, 18033.87it/s]
 22%|
| 91541/414113 [00:05<00:17, 18009.21it/s]
 23%|
| 93360/414113 [00:05<00:17, 18059.34it/s]
 23%|
| 95184/414113 [00:05<00:17, 18112.27it/s]
 23%|
```

```
| 97016/414113 [00:05<00:17, 18173.45it/s]
 24%|
| 98835/414113 [00:05<00:17, 18178.12it/s]
 24%|
| 100653/414113 [00:05<00:17, 18095.61it/s]
 25%|
| 102463/414113 [00:05<00:17, 18003.93it/s]
 25%|
| 104273/414113 [00:05<00:17, 18031.47it/s]
 26%|
| 106077/414113 [00:05<00:17, 17924.44it/s]
 26%|
| 107870/414113 [00:06<00:17, 17866.29it/s]
 26%|
| 109657/414113 [00:06<00:17, 17798.11it/s]
 27%|
| 111448/414113 [00:06<00:16, 17829.17it/s]
 27%|
| 113257/414113 [00:06<00:16, 17905.61it/s]
 28%|
| 115050/414113 [00:06<00:16, 17911.77it/s]
 28%|
| 116851/414113 [00:06<00:16, 17939.08it/s]
 29%|
| 118645/414113 [00:06<00:33, 8866.39it/s]
 29%|
| 120455/414113 [00:07<00:28, 10476.53it/s]
 30%|
| 122276/414113 [00:07<00:24, 12018.22it/s]
 30%|
| 124101/414113 [00:07<00:21, 13399.94it/s]
 30%|
| 125896/414113 [00:07<00:19, 14494.48it/s]
 31%|
| 127707/414113 [00:07<00:18, 15417.30it/s]
 31%|
| 129540/414113 [00:07<00:17, 16194.75it/s]
 32%|
| 131365/414113 [00:07<00:16, 16760.34it/s]
 32%|
| 133164/414113 [00:07<00:16, 17107.36it/s]
 33%|
| 134952/414113 [00:07<00:16, 17230.41it/s]
 33%|
| 136779/414113 [00:07<00:15, 17530.02it/s]
 33%|
| 138577/414113 [00:08<00:15, 17661.33it/s]
 34%|
```

| 140401/414113 [00:08<00:15, 17829.66it/s]
 34%|
| 142204/414113 [00:08<00:15, 17871.12it/s]
 35%|
| 144005/414113 [00:08<00:15, 17889.20it/s]
 35%|
| 145823/414113 [00:08<00:14, 17973.02it/s]
 36%|
| 147628/414113 [00:08<00:14, 17993.65it/s]
 36%|
| 149433/414113 [00:08<00:14, 17992.44it/s]
 37%|
| 151236/414113 [00:08<00:14, 17959.85it/s]
 37%|
| 153035/414113 [00:08<00:14, 17963.21it/s]
 37%|
| 154853/414113 [00:08<00:14, 18027.80it/s]
 38%|
| 156678/414113 [00:09<00:14, 18092.98it/s]
 38%|
| 158495/414113 [00:09<00:14, 18115.03it/s]
 39%|
| 160308/414113 [00:09<00:14, 18093.99it/s]
 39%|
| 162118/414113 [00:09<00:13, 18085.04it/s]
 40%|
| 163957/414113 [00:09<00:13, 18176.17it/s]
 40%|
| 165775/414113 [00:09<00:13, 18097.19it/s]
 40%|
| 167585/414113 [00:09<00:13, 17905.95it/s]
 41%|
| 169377/414113 [00:09<00:13, 17888.71it/s]
 41%|
| 171176/414113 [00:09<00:13, 17917.88it/s]
 42%|
| 172970/414113 [00:09<00:13, 17922.90it/s]
 42%|
| 174763/414113 [00:10<00:13, 17851.98it/s]
 43%|
| 176549/414113 [00:10<00:13, 17753.69it/s]
 43%|
| 178352/414113 [00:10<00:13, 17834.29it/s]
 44%|
| 180141/414113 [00:10<00:13, 17848.59it/s]
 44%|
| 181927/414113 [00:10<00:13, 17839.72it/s]
 44%|

```
| 183726/414113 [00:10<00:12, 17883.06it/s]
 45%|
| 185521/414113 [00:10<00:12, 17902.77it/s]
 45%|
| 187325/414113 [00:10<00:12, 17943.68it/s]
 46%|
| 189120/414113 [00:10<00:12, 17934.64it/s]
 46%|
| 190923/414113 [00:10<00:12, 17961.46it/s]
 47%|
| 192720/414113 [00:11<00:12, 17939.53it/s]
47%|
| 194514/414113 [00:11<00:12, 17928.20it/s]
47%|
| 196328/414113 [00:11<00:12, 17989.67it/s]
 48%|

| 198127/414113 [00:11<00:12, 17985.35it/s]
 48%|

| 199926/414113 [00:11<00:11, 17964.80it/s]
 49%|

| 201723/414113 [00:11<00:11, 17916.81it/s]
 49%|

| 203525/414113 [00:11<00:11, 17946.00it/s]
 50%|

| 205341/414113 [00:11<00:11, 18009.60it/s]
 50%|

| 207142/414113 [00:11<00:11, 17960.58it/s]
 50%|

| 208939/414113 [00:11<00:11, 17931.34it/s]
 51%|

| 210766/414113 [00:12<00:11, 18030.33it/s]
 51%|

| 212598/414113 [00:12<00:11, 18114.40it/s]
 52%|

| 214431/414113 [00:12<00:10, 18177.92it/s]
 52%|

| 216249/414113 [00:12<00:10, 18124.91it/s]
```

```
 53%|

| 218062/414113 [00:12<00:10, 18079.35it/s]
 53%|

| 219894/414113 [00:12<00:10, 18150.05it/s]
 54%|

| 221710/414113 [00:12<00:10, 18113.22it/s]
 54%|

| 223542/414113 [00:12<00:10, 18174.02it/s]
 54%|

| 225360/414113 [00:12<00:10, 18137.81it/s]
 55%|

| 227196/414113 [00:12<00:10, 18201.34it/s]
 55%|

| 229035/414113 [00:13<00:10, 18256.04it/s]
 56%|

| 230861/414113 [00:13<00:10, 18136.86it/s]
 56%|

| 232675/414113 [00:13<00:10, 18036.41it/s]
 57%|

| 234483/414113 [00:13<00:09, 18046.88it/s]
 57%|

| 236291/414113 [00:13<00:09, 18052.91it/s]
 57%|

| 238097/414113 [00:13<00:09, 18050.91it/s]
 58%|

| 239903/414113 [00:13<00:09, 18023.75it/s]
 58%|

| 241706/414113 [00:13<00:09, 17948.94it/s]
 59%|

| 243501/414113 [00:13<00:09, 17906.28it/s]
 59%|

| 245311/414113 [00:13<00:09, 17961.42it/s]
```

```
 60%|

| 247119/414113 [00:14<00:09, 17994.13it/s]
 60%|

| 248927/414113 [00:14<00:09, 18019.02it/s]
 61%|

| 250729/414113 [00:14<00:09, 17972.97it/s]
 61%|

| 252566/414113 [00:14<00:08, 18091.06it/s]
 61%|

| 254378/414113 [00:14<00:08, 18098.51it/s]
 62%|

| 256208/414113 [00:14<00:08, 18158.20it/s]
 62%|

| 258050/414113 [00:14<00:08, 18235.56it/s]
 63%|

| 259874/414113 [00:14<00:08, 18091.22it/s]
 63%|

| 261685/414113 [00:14<00:08, 18096.72it/s]
 64%|

| 263495/414113 [00:14<00:08, 18039.53it/s]
 64%|

| 265300/414113 [00:15<00:08, 17932.98it/s]
 64%|

| 267094/414113 [00:15<00:08, 17810.60it/s]
 65%|

| 268899/414113 [00:15<00:08, 17881.17it/s]
 65%|

| 270694/414113 [00:15<00:08, 17898.71it/s]
 66%|

| 272494/414113 [00:15<00:07, 17927.50it/s]
 66%|

| 274287/414113 [00:15<00:07, 17883.61it/s]
```

67%|

| 276077/414113 [00:15<00:07, 17886.84it/s]
67%|

| 277876/414113 [00:15<00:07, 17917.29it/s]
68%|

| 279668/414113 [00:15<00:07, 17902.01it/s]
68%|

| 281467/414113 [00:16<00:07, 17926.88it/s]
68%|

| 283278/414113 [00:16<00:07, 17979.54it/s]
69%|

| 285076/414113 [00:16<00:07, 17949.04it/s]
69%|

| 286876/414113 [00:16<00:07, 17963.70it/s]
70%|

| 288688/414113 [00:16<00:06, 18009.73it/s]
70%|

| 290506/414113 [00:16<00:06, 18058.88it/s]
71%|

| 292313/414113 [00:16<00:06, 18061.56it/s]
71%|

| 294150/414113 [00:16<00:06, 18152.28it/s]
71%|

| 295966/414113 [00:16<00:06, 18107.37it/s]
72%|

| 297777/414113 [00:16<00:06, 18027.02it/s]
72%|

| 299580/414113 [00:17<00:06, 17985.30it/s]
73%|

| 301379/414113 [00:17<00:06, 17932.80it/s]
73%|

| 303189/414113 [00:17<00:06, 17980.05it/s]

```
 74%|

| 304997/414113 [00:17<00:06, 18008.30it/s]
 74%|

| 306798/414113 [00:17<00:05, 17937.51it/s]
 75%|

| 308592/414113 [00:17<00:05, 17921.74it/s]
 75%|

| 310415/414113 [00:17<00:05, 18012.14it/s]
 75%|

| 312217/414113 [00:17<00:05, 17998.13it/s]
 76%|

| 314021/414113 [00:17<00:05, 18009.48it/s]
 76%|

| 315833/414113 [00:17<00:05, 18039.91it/s]
 77%|

| 317638/414113 [00:18<00:13, 7418.17it/s]
 77%|

| 319442/414113 [00:18<00:10, 9006.88it/s]
 78%|

| 321254/414113 [00:18<00:08, 10610.83it/s]
 78%|

| 323085/414113 [00:18<00:07, 12157.16it/s]
 78%|

| 324911/414113 [00:18<00:06, 13516.86it/s]
 79%|

| 326712/414113 [00:18<00:05, 14601.35it/s]
 79%|

| 328535/414113 [00:19<00:05, 15531.02it/s]
 80%|

| 330311/414113 [00:19<00:05, 16127.83it/s]
 80%|

| 332078/414113 [00:19<00:04, 16534.91it/s]
```

```
 81%|

| 333860/414113 [00:19<00:04, 16898.85it/s]
 81%|

| 335639/414113 [00:19<00:04, 17151.93it/s]
 81%|

| 337418/414113 [00:19<00:04, 17337.55it/s]
 82%|

| 339193/414113 [00:19<00:04, 17431.73it/s]
 82%|

| 340965/414113 [00:19<00:04, 17510.11it/s]
 83%|

| 342736/414113 [00:19<00:04, 17548.24it/s]
 83%|

| 344530/414113 [00:19<00:03, 17662.08it/s]
 84%|

| 346344/414113 [00:20<00:03, 17801.87it/s]
 84%|

| 348132/414113 [00:20<00:03, 17788.50it/s]
 84%|

| 349916/414113 [00:20<00:03, 17771.59it/s]
 85%|

| 351697/414113 [00:20<00:03, 17761.35it/s]
 85%|

| 353479/414113 [00:20<00:03, 17778.27it/s]
 86%|

| 355277/414113 [00:20<00:03, 17838.02it/s]
 86%|

| 357067/414113 [00:20<00:03, 17852.80it/s]
 87%|

| 358854/414113 [00:20<00:03, 17843.22it/s]
 87%|

| 360649/414113 [00:20<00:02, 17873.36it/s]
```

88%|

| 362448/414113 [00:20<00:02, 17906.10it/s]
88%|

| 364239/414113 [00:21<00:02, 17829.42it/s]
88%|

| 366045/414113 [00:21<00:02, 17898.15it/s]
89%|

| 367836/414113 [00:21<00:02, 17831.85it/s]
89%|

| 369631/414113 [00:21<00:02, 17864.86it/s]
90%|

| 371418/414113 [00:21<00:02, 17790.43it/s]
90%|

| 373198/414113 [00:21<00:02, 17785.62it/s]
91%|

| 374989/414113 [00:21<00:02, 17822.67it/s]
91%|

| 376782/414113 [00:21<00:02, 17851.80it/s]
91%|

| 378590/414113 [00:21<00:01, 17917.27it/s]
92%|

| 380382/414113 [00:21<00:01, 17882.18it/s]
92%|

| 382193/414113 [00:22<00:01, 17948.28it/s]
93%|

| 384011/414113 [00:22<00:01, 18017.48it/s]
93%|

| 385813/414113 [00:22<00:01, 17968.62it/s]
94%|

| 387616/414113 [00:22<00:01, 17986.86it/s]
94%|

| 389429/414113 [00:22<00:01, 18029.32it/s]

```
 94%|

| 391232/414113 [00:22<00:01, 17994.20it/s]
 95%|

| 393032/414113 [00:22<00:01, 17922.63it/s]
 95%|

| 394825/414113 [00:22<00:01, 17849.31it/s]
 96%|

| 396611/414113 [00:22<00:00, 17850.06it/s]
 96%|

| 398410/414113 [00:23<00:00, 17889.98it/s]
 97%|

| 400200/414113 [00:23<00:00, 17853.00it/s]
 97%|

    | 401986/414113 [00:23<00:00, 17784.18it/s]
 98%|

    | 403765/414113 [00:23<00:00, 17756.18it/s]
 98%|

    | 405551/414113 [00:23<00:00, 17786.11it/s]
 98%|

   | 407350/414113 [00:23<00:00, 17844.87it/s]
 99%|

   | 409153/414113 [00:23<00:00, 17899.34it/s]
 99%|

   | 410964/414113 [00:23<00:00, 17959.73it/s]
100%|

   | 414113/414113 [00:23<00:00, 17340.64it/s]
```

Step 2: Train your Model

Once you have executed the code cell in **Step 1**, the training procedure below should run without issue.

It is completely fine to leave the code cell below as-is without modifications to train your model. However, if you would like to modify the code used to train the model below, you must ensure that your changes are easily parsed by your reviewer. In other words, make sure to provide appropriate comments to describe how your code works!

You may find it useful to load saved weights to resume training. In that case, note the names of the files containing the encoder and decoder weights that you'd like to load (`encoder_file` and `decoder_file`). Then you can load the weights by using the lines below:

```
# Load pre-trained weights before resuming training.
encoder.load_state_dict(torch.load(os.path.join('./models', encoder_file)))
decoder.load_state_dict(torch.load(os.path.join('./models', decoder_file)))
```

While trying out parameters, make sure to take extensive notes and record the settings that you used in your various training runs. In particular, you don't want to encounter a situation where you've trained a model for several hours but can't remember what settings you used :).

1.1.9 A Note on Tuning Hyperparameters

To figure out how well your model is doing, you can look at how the training loss and perplexity evolve during training - and for the purposes of this project, you are encouraged to amend the hyperparameters based on this information.

However, this will not tell you if your model is overfitting to the training data, and, unfortunately, overfitting is a problem that is commonly encountered when training image captioning models.

For this project, you need not worry about overfitting. **This project does not have strict requirements regarding the performance of your model**, and you just need to demonstrate that your model has learned ***something*** when you generate captions on the test data. For now, we strongly encourage you to train your model for the suggested 3 epochs without worrying about performance; then, you should immediately transition to the next notebook in the sequence (**3_Inference.ipynb**) to see how your model performs on the test data. If your model needs to be changed, you can come back to this notebook, amend hyperparameters (if necessary), and re-train the model.

That said, if you would like to go above and beyond in this project, you can read about some approaches to minimizing overfitting in section 4.3.1 of this paper. In the next (optional) step of this notebook, we provide some guidance for assessing the performance on the validation dataset.

[12]:
```
import torch.utils.data as data
import numpy as np
import os
import requests
import time

# Open the training log file.
f = open(log_file, 'w')

old_time = time.time()
#response = requests.request("GET",
#                            "http://metadata.google.internal/computeMetadata/
 ↪v1/instance/attributes/keep_alive_token",
#                            headers={"Metadata-Flavor":"Google"})

for epoch in range(1, num_epochs+1):
```

```
    for i_step in range(1, total_step+1):

        #if time.time() - old_time > 60:
        #    old_time = time.time()
        #    requests.request("POST",
        #                     "https://nebula.udacity.com/api/v1/remote/
↪keep-alive",
        #                     headers={'Authorization': "STAR " + response.
↪text})

        # Randomly sample a caption length, and sample indices with that length.
        indices = data_loader.dataset.get_train_indices()
        # Create and assign a batch sampler to retrieve a batch with the␣
↪sampled indices.
        new_sampler = data.sampler.SubsetRandomSampler(indices=indices)
        data_loader.batch_sampler.sampler = new_sampler

        # Obtain the batch.
        images, captions = next(iter(data_loader))

        # Move batch of images and captions to GPU if CUDA is available.
        images = images.to(device)
        captions = captions.to(device)

        # Zero the gradients.
        decoder.zero_grad()
        encoder.zero_grad()

        # Pass the inputs through the CNN-RNN model.
        features = encoder(images)
        outputs = decoder(features, captions)

        # Calculate the batch loss.
        loss = criterion(outputs.view(-1, vocab_size), captions.view(-1))

        # Backward pass.
        loss.backward()

        # Update the parameters in the optimizer.
        optimizer.step()

        # Get training statistics.
        stats = 'Epoch [%d/%d], Step [%d/%d], Loss: %.4f, Perplexity: %5.4f' %␣
↪(epoch, num_epochs, i_step, total_step, loss.item(), np.exp(loss.item()))

        # Print training statistics (on same line).
```

```
        print('\r' + stats, end="")
        sys.stdout.flush()

        # Print training statistics to file.
        f.write(stats + '\n')
        f.flush()

        # Print training statistics (on different line).
        if i_step % print_every == 0:
            print('\r' + stats)

    # Save the weights.
    if epoch % save_every == 0:
        torch.save(decoder.state_dict(), os.path.join('./models', 'decoder-%d.
 ↪pkl' % epoch))
        torch.save(encoder.state_dict(), os.path.join('./models', 'encoder-%d.
 ↪pkl' % epoch))

# Close the training log file.
f.close()
```

```
Epoch [1/1], Step [100/41412], Loss: 4.2345, Perplexity: 69.02683
Epoch [1/1], Step [200/41412], Loss: 4.3270, Perplexity: 75.71703
Epoch [1/1], Step [300/41412], Loss: 3.4750, Perplexity: 32.29716
Epoch [1/1], Step [400/41412], Loss: 4.0520, Perplexity: 57.51179
Epoch [1/1], Step [500/41412], Loss: 3.5070, Perplexity: 33.34873
Epoch [1/1], Step [600/41412], Loss: 3.5980, Perplexity: 36.52555
Epoch [1/1], Step [700/41412], Loss: 3.7124, Perplexity: 40.9513
Epoch [1/1], Step [800/41412], Loss: 2.8845, Perplexity: 17.8944
Epoch [1/1], Step [900/41412], Loss: 3.2459, Perplexity: 25.6852
Epoch [1/1], Step [1000/41412], Loss: 3.5327, Perplexity: 34.2166
Epoch [1/1], Step [1100/41412], Loss: 3.2321, Perplexity: 25.3333
Epoch [1/1], Step [1200/41412], Loss: 3.1583, Perplexity: 23.5295
Epoch [1/1], Step [1300/41412], Loss: 3.4732, Perplexity: 32.2406
Epoch [1/1], Step [1400/41412], Loss: 3.5148, Perplexity: 33.6100
Epoch [1/1], Step [1500/41412], Loss: 2.6269, Perplexity: 13.8313
Epoch [1/1], Step [1600/41412], Loss: 2.7274, Perplexity: 15.2938
Epoch [1/1], Step [1700/41412], Loss: 2.5649, Perplexity: 12.9997
Epoch [1/1], Step [1800/41412], Loss: 2.9912, Perplexity: 19.90949
Epoch [1/1], Step [1900/41412], Loss: 3.6450, Perplexity: 38.2844
Epoch [1/1], Step [2000/41412], Loss: 2.9031, Perplexity: 18.2314
Epoch [1/1], Step [2100/41412], Loss: 3.2184, Perplexity: 24.9870
Epoch [1/1], Step [2200/41412], Loss: 2.9897, Perplexity: 19.8806
Epoch [1/1], Step [2300/41412], Loss: 3.3926, Perplexity: 29.7438
Epoch [1/1], Step [2400/41412], Loss: 2.8217, Perplexity: 16.8058
Epoch [1/1], Step [2500/41412], Loss: 2.3323, Perplexity: 10.3020
Epoch [1/1], Step [2600/41412], Loss: 3.0511, Perplexity: 21.1391
```

```
Epoch [1/1], Step [2700/41412], Loss: 2.8234, Perplexity: 16.8341
Epoch [1/1], Step [2800/41412], Loss: 2.9651, Perplexity: 19.39680
Epoch [1/1], Step [2900/41412], Loss: 2.6127, Perplexity: 13.6354
Epoch [1/1], Step [3000/41412], Loss: 2.2675, Perplexity: 9.65501
Epoch [1/1], Step [3100/41412], Loss: 3.0904, Perplexity: 21.9857
Epoch [1/1], Step [3200/41412], Loss: 3.3145, Perplexity: 27.5095
Epoch [1/1], Step [3300/41412], Loss: 3.2225, Perplexity: 25.0914
Epoch [1/1], Step [3400/41412], Loss: 3.1256, Perplexity: 22.7729
Epoch [1/1], Step [3500/41412], Loss: 2.8475, Perplexity: 17.2441
Epoch [1/1], Step [3600/41412], Loss: 3.5544, Perplexity: 34.9669
Epoch [1/1], Step [3700/41412], Loss: 2.6883, Perplexity: 14.7062
Epoch [1/1], Step [3800/41412], Loss: 3.1020, Perplexity: 22.2434
Epoch [1/1], Step [3900/41412], Loss: 3.1398, Perplexity: 23.09957
Epoch [1/1], Step [4000/41412], Loss: 3.4279, Perplexity: 30.8105
Epoch [1/1], Step [4100/41412], Loss: 2.9569, Perplexity: 19.2383
Epoch [1/1], Step [4200/41412], Loss: 3.3775, Perplexity: 29.2973
Epoch [1/1], Step [4300/41412], Loss: 2.5156, Perplexity: 12.3746
Epoch [1/1], Step [4400/41412], Loss: 2.7862, Perplexity: 16.2198
Epoch [1/1], Step [4500/41412], Loss: 2.2126, Perplexity: 9.13970
Epoch [1/1], Step [4600/41412], Loss: 3.1162, Perplexity: 22.5599
Epoch [1/1], Step [4700/41412], Loss: 2.9170, Perplexity: 18.4858
Epoch [1/1], Step [4800/41412], Loss: 2.7882, Perplexity: 16.2520
Epoch [1/1], Step [4900/41412], Loss: 2.4391, Perplexity: 11.4622
Epoch [1/1], Step [5000/41412], Loss: 2.9029, Perplexity: 18.2262
Epoch [1/1], Step [5100/41412], Loss: 2.6389, Perplexity: 13.9982
Epoch [1/1], Step [5200/41412], Loss: 2.9743, Perplexity: 19.5766
Epoch [1/1], Step [5300/41412], Loss: 2.5832, Perplexity: 13.2397
Epoch [1/1], Step [5400/41412], Loss: 2.7555, Perplexity: 15.7284
Epoch [1/1], Step [5500/41412], Loss: 2.0315, Perplexity: 7.62584
Epoch [1/1], Step [5600/41412], Loss: 3.1634, Perplexity: 23.6507
Epoch [1/1], Step [5700/41412], Loss: 2.8374, Perplexity: 17.0710
Epoch [1/1], Step [5800/41412], Loss: 3.3649, Perplexity: 28.9318
Epoch [1/1], Step [5900/41412], Loss: 2.8076, Perplexity: 16.5701
Epoch [1/1], Step [6000/41412], Loss: 2.7135, Perplexity: 15.0821
Epoch [1/1], Step [6100/41412], Loss: 2.6901, Perplexity: 14.7325
Epoch [1/1], Step [6200/41412], Loss: 3.0982, Perplexity: 22.1580
Epoch [1/1], Step [6300/41412], Loss: 2.2835, Perplexity: 9.81123
Epoch [1/1], Step [6400/41412], Loss: 2.1984, Perplexity: 9.01054
Epoch [1/1], Step [6500/41412], Loss: 2.1995, Perplexity: 9.02042
Epoch [1/1], Step [6600/41412], Loss: 2.6873, Perplexity: 14.6917
Epoch [1/1], Step [6700/41412], Loss: 2.8606, Perplexity: 17.4720
Epoch [1/1], Step [6800/41412], Loss: 2.4572, Perplexity: 11.6718
Epoch [1/1], Step [6900/41412], Loss: 3.5079, Perplexity: 33.37860
Epoch [1/1], Step [7000/41412], Loss: 2.6088, Perplexity: 13.58327
Epoch [1/1], Step [7100/41412], Loss: 2.3611, Perplexity: 10.6023
Epoch [1/1], Step [7200/41412], Loss: 2.9342, Perplexity: 18.8059
Epoch [1/1], Step [7300/41412], Loss: 2.8820, Perplexity: 17.8494
Epoch [1/1], Step [7400/41412], Loss: 2.2418, Perplexity: 9.41077
```

```
Epoch [1/1], Step [7500/41412], Loss: 2.3721, Perplexity: 10.7195
Epoch [1/1], Step [7600/41412], Loss: 2.1241, Perplexity: 8.36504
Epoch [1/1], Step [7700/41412], Loss: 2.2765, Perplexity: 9.74224
Epoch [1/1], Step [7800/41412], Loss: 2.5722, Perplexity: 13.0942
Epoch [1/1], Step [7900/41412], Loss: 2.7545, Perplexity: 15.7128
Epoch [1/1], Step [8000/41412], Loss: 2.3548, Perplexity: 10.5359
Epoch [1/1], Step [8100/41412], Loss: 2.7079, Perplexity: 14.9983
Epoch [1/1], Step [8200/41412], Loss: 2.3472, Perplexity: 10.4562
Epoch [1/1], Step [8300/41412], Loss: 2.6449, Perplexity: 14.0825
Epoch [1/1], Step [8400/41412], Loss: 2.4387, Perplexity: 11.4587
Epoch [1/1], Step [8500/41412], Loss: 3.2756, Perplexity: 26.4581
Epoch [1/1], Step [8600/41412], Loss: 2.8290, Perplexity: 16.9279
Epoch [1/1], Step [8700/41412], Loss: 3.1885, Perplexity: 24.2511
Epoch [1/1], Step [8800/41412], Loss: 2.6253, Perplexity: 13.80933
Epoch [1/1], Step [8900/41412], Loss: 3.2530, Perplexity: 25.8683
Epoch [1/1], Step [9000/41412], Loss: 2.6461, Perplexity: 14.0992
Epoch [1/1], Step [9100/41412], Loss: 2.5844, Perplexity: 13.2551
Epoch [1/1], Step [9200/41412], Loss: 2.0402, Perplexity: 7.69203
Epoch [1/1], Step [9300/41412], Loss: 2.9810, Perplexity: 19.70785
Epoch [1/1], Step [9400/41412], Loss: 2.2967, Perplexity: 9.94096
Epoch [1/1], Step [9500/41412], Loss: 2.5944, Perplexity: 13.3888
Epoch [1/1], Step [9600/41412], Loss: 2.6767, Perplexity: 14.53732
Epoch [1/1], Step [9700/41412], Loss: 4.4122, Perplexity: 82.4501
Epoch [1/1], Step [9800/41412], Loss: 2.1528, Perplexity: 8.60865
Epoch [1/1], Step [9900/41412], Loss: 2.7132, Perplexity: 15.0782
Epoch [1/1], Step [10000/41412], Loss: 2.4806, Perplexity: 11.9488
Epoch [1/1], Step [10100/41412], Loss: 1.8365, Perplexity: 6.27440
Epoch [1/1], Step [10200/41412], Loss: 2.1647, Perplexity: 8.71210
Epoch [1/1], Step [10300/41412], Loss: 2.3071, Perplexity: 10.0454
Epoch [1/1], Step [10400/41412], Loss: 2.3696, Perplexity: 10.6934
Epoch [1/1], Step [10500/41412], Loss: 2.3348, Perplexity: 10.3276
Epoch [1/1], Step [10600/41412], Loss: 3.3436, Perplexity: 28.3219
Epoch [1/1], Step [10700/41412], Loss: 2.4591, Perplexity: 11.6939
Epoch [1/1], Step [10800/41412], Loss: 2.7832, Perplexity: 16.1712
Epoch [1/1], Step [10900/41412], Loss: 2.6539, Perplexity: 14.2094
Epoch [1/1], Step [11000/41412], Loss: 2.5179, Perplexity: 12.4027
Epoch [1/1], Step [11100/41412], Loss: 2.1497, Perplexity: 8.58230
Epoch [1/1], Step [11200/41412], Loss: 2.2001, Perplexity: 9.026359
Epoch [1/1], Step [11300/41412], Loss: 2.7015, Perplexity: 14.9019
Epoch [1/1], Step [11400/41412], Loss: 2.9303, Perplexity: 18.7324
Epoch [1/1], Step [11500/41412], Loss: 3.0500, Perplexity: 21.1157
Epoch [1/1], Step [11600/41412], Loss: 2.5492, Perplexity: 12.7967
Epoch [1/1], Step [11700/41412], Loss: 2.2620, Perplexity: 9.60275
Epoch [1/1], Step [11800/41412], Loss: 3.5452, Perplexity: 34.6470
Epoch [1/1], Step [11900/41412], Loss: 2.3905, Perplexity: 10.9188
Epoch [1/1], Step [12000/41412], Loss: 2.3142, Perplexity: 10.1173
Epoch [1/1], Step [12100/41412], Loss: 2.2261, Perplexity: 9.26389
Epoch [1/1], Step [12200/41412], Loss: 2.9741, Perplexity: 19.5727
```

```
Epoch [1/1], Step [12300/41412], Loss: 2.4848, Perplexity: 11.9985
Epoch [1/1], Step [12400/41412], Loss: 3.0757, Perplexity: 21.6645
Epoch [1/1], Step [12500/41412], Loss: 2.2434, Perplexity: 9.42561
Epoch [1/1], Step [12600/41412], Loss: 2.3487, Perplexity: 10.4722
Epoch [1/1], Step [12700/41412], Loss: 2.8038, Perplexity: 16.5076
Epoch [1/1], Step [12800/41412], Loss: 2.4396, Perplexity: 11.4683
Epoch [1/1], Step [12900/41412], Loss: 2.1559, Perplexity: 8.63603
Epoch [1/1], Step [13000/41412], Loss: 3.1821, Perplexity: 24.0962
Epoch [1/1], Step [13100/41412], Loss: 2.5530, Perplexity: 12.8454
Epoch [1/1], Step [13200/41412], Loss: 2.6427, Perplexity: 14.0513
Epoch [1/1], Step [13300/41412], Loss: 2.4969, Perplexity: 12.1446
Epoch [1/1], Step [13400/41412], Loss: 2.5521, Perplexity: 12.8338
Epoch [1/1], Step [13500/41412], Loss: 2.2491, Perplexity: 9.47900
Epoch [1/1], Step [13600/41412], Loss: 1.9513, Perplexity: 7.03784
Epoch [1/1], Step [13700/41412], Loss: 2.6578, Perplexity: 14.2643
Epoch [1/1], Step [13800/41412], Loss: 2.1111, Perplexity: 8.25763
Epoch [1/1], Step [13900/41412], Loss: 2.1971, Perplexity: 8.99875
Epoch [1/1], Step [14000/41412], Loss: 2.7119, Perplexity: 15.0571
Epoch [1/1], Step [14100/41412], Loss: 2.1214, Perplexity: 8.34248
Epoch [1/1], Step [14200/41412], Loss: 2.2231, Perplexity: 9.23617
Epoch [1/1], Step [14300/41412], Loss: 2.2280, Perplexity: 9.28090
Epoch [1/1], Step [14400/41412], Loss: 2.6029, Perplexity: 13.5031
Epoch [1/1], Step [14500/41412], Loss: 2.4591, Perplexity: 11.6946
Epoch [1/1], Step [14600/41412], Loss: 2.6229, Perplexity: 13.7760
Epoch [1/1], Step [14700/41412], Loss: 2.4555, Perplexity: 11.6517
Epoch [1/1], Step [14800/41412], Loss: 2.1364, Perplexity: 8.46911
Epoch [1/1], Step [14900/41412], Loss: 2.5010, Perplexity: 12.1944
Epoch [1/1], Step [15000/41412], Loss: 2.0889, Perplexity: 8.07570
Epoch [1/1], Step [15100/41412], Loss: 2.4574, Perplexity: 11.6739
Epoch [1/1], Step [15200/41412], Loss: 3.3795, Perplexity: 29.3567
Epoch [1/1], Step [15300/41412], Loss: 2.7870, Perplexity: 16.2325
Epoch [1/1], Step [15400/41412], Loss: 1.8103, Perplexity: 6.11225
Epoch [1/1], Step [15500/41412], Loss: 2.6663, Perplexity: 14.3872
Epoch [1/1], Step [15600/41412], Loss: 2.6829, Perplexity: 14.6271
Epoch [1/1], Step [15700/41412], Loss: 2.3384, Perplexity: 10.3646
Epoch [1/1], Step [15800/41412], Loss: 2.1767, Perplexity: 8.81729
Epoch [1/1], Step [15900/41412], Loss: 2.0904, Perplexity: 8.08834
Epoch [1/1], Step [16000/41412], Loss: 2.4323, Perplexity: 11.3847
Epoch [1/1], Step [16100/41412], Loss: 2.5874, Perplexity: 13.2957
Epoch [1/1], Step [16200/41412], Loss: 1.9664, Perplexity: 7.14517
Epoch [1/1], Step [16300/41412], Loss: 2.5090, Perplexity: 12.2921
Epoch [1/1], Step [16400/41412], Loss: 2.4991, Perplexity: 12.1714
Epoch [1/1], Step [16500/41412], Loss: 2.3877, Perplexity: 10.8883
Epoch [1/1], Step [16600/41412], Loss: 2.2630, Perplexity: 9.61230
Epoch [1/1], Step [16700/41412], Loss: 2.6102, Perplexity: 13.6016
Epoch [1/1], Step [16800/41412], Loss: 2.4637, Perplexity: 11.7485
Epoch [1/1], Step [16900/41412], Loss: 2.4887, Perplexity: 12.0460
Epoch [1/1], Step [17000/41412], Loss: 2.3008, Perplexity: 9.98176
```

```
Epoch [1/1], Step [17100/41412], Loss: 2.2598, Perplexity: 9.58109
Epoch [1/1], Step [17200/41412], Loss: 2.3336, Perplexity: 10.3147
Epoch [1/1], Step [17300/41412], Loss: 2.7139, Perplexity: 15.0876
Epoch [1/1], Step [17400/41412], Loss: 2.4845, Perplexity: 11.99500
Epoch [1/1], Step [17500/41412], Loss: 2.7437, Perplexity: 15.5439
Epoch [1/1], Step [17600/41412], Loss: 2.0321, Perplexity: 7.63049
Epoch [1/1], Step [17700/41412], Loss: 2.1440, Perplexity: 8.53325
Epoch [1/1], Step [17800/41412], Loss: 2.4575, Perplexity: 11.6754
Epoch [1/1], Step [17900/41412], Loss: 2.9434, Perplexity: 18.9811
Epoch [1/1], Step [18000/41412], Loss: 2.4173, Perplexity: 11.2158
Epoch [1/1], Step [18100/41412], Loss: 2.2562, Perplexity: 9.54706
Epoch [1/1], Step [18200/41412], Loss: 2.1352, Perplexity: 8.45865
Epoch [1/1], Step [18300/41412], Loss: 2.2795, Perplexity: 9.77173
Epoch [1/1], Step [18400/41412], Loss: 2.0743, Perplexity: 7.95915
Epoch [1/1], Step [18500/41412], Loss: 2.4009, Perplexity: 11.0334
Epoch [1/1], Step [18600/41412], Loss: 1.9947, Perplexity: 7.34996
Epoch [1/1], Step [18700/41412], Loss: 2.2482, Perplexity: 9.47083
Epoch [1/1], Step [18800/41412], Loss: 2.7383, Perplexity: 15.4602
Epoch [1/1], Step [18900/41412], Loss: 2.3079, Perplexity: 10.0538
Epoch [1/1], Step [19000/41412], Loss: 2.8255, Perplexity: 16.8700
Epoch [1/1], Step [19100/41412], Loss: 2.5174, Perplexity: 12.3962
Epoch [1/1], Step [19200/41412], Loss: 2.3018, Perplexity: 9.99187
Epoch [1/1], Step [19300/41412], Loss: 1.6960, Perplexity: 5.45228
Epoch [1/1], Step [19400/41412], Loss: 3.2233, Perplexity: 25.1097
Epoch [1/1], Step [19500/41412], Loss: 2.2474, Perplexity: 9.46357
Epoch [1/1], Step [19600/41412], Loss: 2.8693, Perplexity: 17.6251
Epoch [1/1], Step [19700/41412], Loss: 2.4894, Perplexity: 12.0544
Epoch [1/1], Step [19800/41412], Loss: 2.1486, Perplexity: 8.57250
Epoch [1/1], Step [19900/41412], Loss: 1.8573, Perplexity: 6.40662
Epoch [1/1], Step [20000/41412], Loss: 2.1967, Perplexity: 8.99571
Epoch [1/1], Step [20100/41412], Loss: 2.5468, Perplexity: 12.7661
Epoch [1/1], Step [20200/41412], Loss: 2.3628, Perplexity: 10.6211
Epoch [1/1], Step [20300/41412], Loss: 2.4361, Perplexity: 11.4288
Epoch [1/1], Step [20400/41412], Loss: 2.3939, Perplexity: 10.9559
Epoch [1/1], Step [20500/41412], Loss: 2.2321, Perplexity: 9.31915
Epoch [1/1], Step [20600/41412], Loss: 2.3035, Perplexity: 10.0093
Epoch [1/1], Step [20700/41412], Loss: 2.4137, Perplexity: 11.1747
Epoch [1/1], Step [20800/41412], Loss: 2.4206, Perplexity: 11.2530
Epoch [1/1], Step [20900/41412], Loss: 2.1842, Perplexity: 8.88377
Epoch [1/1], Step [21000/41412], Loss: 2.5299, Perplexity: 12.5522
Epoch [1/1], Step [21100/41412], Loss: 2.0497, Perplexity: 7.76560
Epoch [1/1], Step [21200/41412], Loss: 1.8699, Perplexity: 6.48757
Epoch [1/1], Step [21300/41412], Loss: 2.8732, Perplexity: 17.6927
Epoch [1/1], Step [21400/41412], Loss: 2.4283, Perplexity: 11.3395
Epoch [1/1], Step [21500/41412], Loss: 1.8955, Perplexity: 6.655879
Epoch [1/1], Step [21600/41412], Loss: 2.0500, Perplexity: 7.76819
Epoch [1/1], Step [21700/41412], Loss: 2.0384, Perplexity: 7.67824
Epoch [1/1], Step [21800/41412], Loss: 1.7400, Perplexity: 5.69745
```

```
Epoch [1/1], Step [21900/41412], Loss: 2.1628, Perplexity: 8.69568
Epoch [1/1], Step [22000/41412], Loss: 2.0063, Perplexity: 7.43610
Epoch [1/1], Step [22100/41412], Loss: 1.9544, Perplexity: 7.05969
Epoch [1/1], Step [22200/41412], Loss: 2.4306, Perplexity: 11.3660
Epoch [1/1], Step [22300/41412], Loss: 2.3800, Perplexity: 10.8047
Epoch [1/1], Step [22400/41412], Loss: 2.2854, Perplexity: 9.83010
Epoch [1/1], Step [22500/41412], Loss: 3.0032, Perplexity: 20.1498
Epoch [1/1], Step [22600/41412], Loss: 2.2988, Perplexity: 9.96232
Epoch [1/1], Step [22700/41412], Loss: 2.6431, Perplexity: 14.0570
Epoch [1/1], Step [22800/41412], Loss: 2.4911, Perplexity: 12.0751
Epoch [1/1], Step [22900/41412], Loss: 2.2549, Perplexity: 9.53416
Epoch [1/1], Step [23000/41412], Loss: 2.4513, Perplexity: 11.6036
Epoch [1/1], Step [23100/41412], Loss: 2.1517, Perplexity: 8.59961
Epoch [1/1], Step [23200/41412], Loss: 3.3619, Perplexity: 28.8441
Epoch [1/1], Step [23300/41412], Loss: 2.3564, Perplexity: 10.5531
Epoch [1/1], Step [23400/41412], Loss: 2.0761, Perplexity: 7.97351
Epoch [1/1], Step [23500/41412], Loss: 2.1701, Perplexity: 8.75893
Epoch [1/1], Step [23600/41412], Loss: 2.1420, Perplexity: 8.51623
Epoch [1/1], Step [23700/41412], Loss: 2.3580, Perplexity: 10.5703
Epoch [1/1], Step [23800/41412], Loss: 2.8624, Perplexity: 17.5040
Epoch [1/1], Step [23900/41412], Loss: 2.1601, Perplexity: 8.67228
Epoch [1/1], Step [24000/41412], Loss: 2.0483, Perplexity: 7.75507
Epoch [1/1], Step [24100/41412], Loss: 2.0168, Perplexity: 7.51408
Epoch [1/1], Step [24200/41412], Loss: 2.6449, Perplexity: 14.0825
Epoch [1/1], Step [24300/41412], Loss: 2.4012, Perplexity: 11.0366
Epoch [1/1], Step [24400/41412], Loss: 2.6452, Perplexity: 14.0859
Epoch [1/1], Step [24500/41412], Loss: 2.6540, Perplexity: 14.2114
Epoch [1/1], Step [24600/41412], Loss: 2.3806, Perplexity: 10.8109
Epoch [1/1], Step [24700/41412], Loss: 2.3502, Perplexity: 10.4881
Epoch [1/1], Step [24800/41412], Loss: 2.2863, Perplexity: 9.83852
Epoch [1/1], Step [24900/41412], Loss: 2.3885, Perplexity: 10.8977
Epoch [1/1], Step [25000/41412], Loss: 2.6866, Perplexity: 14.6815
Epoch [1/1], Step [25100/41412], Loss: 2.7337, Perplexity: 15.3896
Epoch [1/1], Step [25200/41412], Loss: 2.4709, Perplexity: 11.8333
Epoch [1/1], Step [25300/41412], Loss: 2.8935, Perplexity: 18.0567
Epoch [1/1], Step [25400/41412], Loss: 2.4109, Perplexity: 11.1445
Epoch [1/1], Step [25500/41412], Loss: 2.5319, Perplexity: 12.5778
Epoch [1/1], Step [25600/41412], Loss: 2.7509, Perplexity: 15.6575
Epoch [1/1], Step [25700/41412], Loss: 2.6914, Perplexity: 14.7526
Epoch [1/1], Step [25800/41412], Loss: 2.2250, Perplexity: 9.25354
Epoch [1/1], Step [25900/41412], Loss: 2.2397, Perplexity: 9.39011
Epoch [1/1], Step [26000/41412], Loss: 1.9295, Perplexity: 6.88592
Epoch [1/1], Step [26100/41412], Loss: 2.5813, Perplexity: 13.2144
Epoch [1/1], Step [26200/41412], Loss: 2.3018, Perplexity: 9.99241
Epoch [1/1], Step [26300/41412], Loss: 2.3180, Perplexity: 10.1551
Epoch [1/1], Step [26400/41412], Loss: 2.7216, Perplexity: 15.2043
Epoch [1/1], Step [26500/41412], Loss: 2.2249, Perplexity: 9.25295
Epoch [1/1], Step [26600/41412], Loss: 2.2724, Perplexity: 9.70297
```

```
Epoch [1/1], Step [26700/41412], Loss: 1.9138, Perplexity: 6.77890
Epoch [1/1], Step [26800/41412], Loss: 2.7595, Perplexity: 15.7913
Epoch [1/1], Step [26900/41412], Loss: 2.7869, Perplexity: 16.2313
Epoch [1/1], Step [27000/41412], Loss: 2.3032, Perplexity: 10.0059
Epoch [1/1], Step [27100/41412], Loss: 2.2373, Perplexity: 9.36802
Epoch [1/1], Step [27200/41412], Loss: 2.4111, Perplexity: 11.1458
Epoch [1/1], Step [27300/41412], Loss: 2.1488, Perplexity: 8.57476
Epoch [1/1], Step [27400/41412], Loss: 2.3136, Perplexity: 10.1110
Epoch [1/1], Step [27500/41412], Loss: 2.4621, Perplexity: 11.7295
Epoch [1/1], Step [27600/41412], Loss: 2.0178, Perplexity: 7.52160
Epoch [1/1], Step [27700/41412], Loss: 2.0183, Perplexity: 7.52541
Epoch [1/1], Step [27800/41412], Loss: 2.6725, Perplexity: 14.4759
Epoch [1/1], Step [27900/41412], Loss: 2.3243, Perplexity: 10.2192
Epoch [1/1], Step [28000/41412], Loss: 2.4651, Perplexity: 11.7642
Epoch [1/1], Step [28100/41412], Loss: 2.3345, Perplexity: 10.3242
Epoch [1/1], Step [28200/41412], Loss: 2.0323, Perplexity: 7.63184
Epoch [1/1], Step [28300/41412], Loss: 2.1901, Perplexity: 8.93657
Epoch [1/1], Step [28400/41412], Loss: 1.8498, Perplexity: 6.35844
Epoch [1/1], Step [28500/41412], Loss: 2.4876, Perplexity: 12.0324
Epoch [1/1], Step [28600/41412], Loss: 2.4744, Perplexity: 11.8750
Epoch [1/1], Step [28700/41412], Loss: 2.1797, Perplexity: 8.84331
Epoch [1/1], Step [28800/41412], Loss: 1.9512, Perplexity: 7.03723
Epoch [1/1], Step [28900/41412], Loss: 2.5952, Perplexity: 13.3995
Epoch [1/1], Step [29000/41412], Loss: 2.1845, Perplexity: 8.88653
Epoch [1/1], Step [29100/41412], Loss: 2.6488, Perplexity: 14.1373
Epoch [1/1], Step [29200/41412], Loss: 2.7346, Perplexity: 15.4035
Epoch [1/1], Step [29300/41412], Loss: 2.0987, Perplexity: 8.15538
Epoch [1/1], Step [29400/41412], Loss: 2.4372, Perplexity: 11.4411
Epoch [1/1], Step [29500/41412], Loss: 2.2585, Perplexity: 9.56888
Epoch [1/1], Step [29600/41412], Loss: 1.9708, Perplexity: 7.176146
Epoch [1/1], Step [29700/41412], Loss: 2.2131, Perplexity: 9.14440
Epoch [1/1], Step [29800/41412], Loss: 2.3101, Perplexity: 10.0750
Epoch [1/1], Step [29900/41412], Loss: 2.6722, Perplexity: 14.4712
Epoch [1/1], Step [30000/41412], Loss: 2.6577, Perplexity: 14.2629
Epoch [1/1], Step [30100/41412], Loss: 2.4141, Perplexity: 11.1793
Epoch [1/1], Step [30200/41412], Loss: 2.8234, Perplexity: 16.8346
Epoch [1/1], Step [30300/41412], Loss: 2.3444, Perplexity: 10.4272
Epoch [1/1], Step [30400/41412], Loss: 2.7279, Perplexity: 15.3001
Epoch [1/1], Step [30500/41412], Loss: 2.4291, Perplexity: 11.3484
Epoch [1/1], Step [30600/41412], Loss: 2.3097, Perplexity: 10.0712
Epoch [1/1], Step [30700/41412], Loss: 2.4684, Perplexity: 11.8035
Epoch [1/1], Step [30800/41412], Loss: 2.6246, Perplexity: 13.7984
Epoch [1/1], Step [30900/41412], Loss: 2.2968, Perplexity: 9.94248
Epoch [1/1], Step [31000/41412], Loss: 1.9035, Perplexity: 6.70922
Epoch [1/1], Step [31100/41412], Loss: 2.7335, Perplexity: 15.3872
Epoch [1/1], Step [31200/41412], Loss: 2.2243, Perplexity: 9.24746
Epoch [1/1], Step [31300/41412], Loss: 2.3462, Perplexity: 10.4457
Epoch [1/1], Step [31400/41412], Loss: 2.6156, Perplexity: 13.6761
```

```
Epoch [1/1], Step [31500/41412], Loss: 2.3379, Perplexity: 10.3594
Epoch [1/1], Step [31600/41412], Loss: 2.3968, Perplexity: 10.9881
Epoch [1/1], Step [31700/41412], Loss: 2.1161, Perplexity: 8.29867
Epoch [1/1], Step [31800/41412], Loss: 2.1761, Perplexity: 8.81224
Epoch [1/1], Step [31900/41412], Loss: 2.8227, Perplexity: 16.8219
Epoch [1/1], Step [32000/41412], Loss: 2.3075, Perplexity: 10.0497
Epoch [1/1], Step [32100/41412], Loss: 2.6873, Perplexity: 14.6919
Epoch [1/1], Step [32200/41412], Loss: 3.4497, Perplexity: 31.4911
Epoch [1/1], Step [32300/41412], Loss: 2.2389, Perplexity: 9.38324
Epoch [1/1], Step [32400/41412], Loss: 2.4003, Perplexity: 11.0261
Epoch [1/1], Step [32500/41412], Loss: 3.0131, Perplexity: 20.3496
Epoch [1/1], Step [32600/41412], Loss: 2.9272, Perplexity: 18.6747
Epoch [1/1], Step [32700/41412], Loss: 2.1971, Perplexity: 8.99920
Epoch [1/1], Step [32800/41412], Loss: 1.9386, Perplexity: 6.94900
Epoch [1/1], Step [32900/41412], Loss: 2.5447, Perplexity: 12.7398
Epoch [1/1], Step [33000/41412], Loss: 2.7981, Perplexity: 16.4133
Epoch [1/1], Step [33100/41412], Loss: 2.7982, Perplexity: 16.4147
Epoch [1/1], Step [33200/41412], Loss: 2.4566, Perplexity: 11.6646
Epoch [1/1], Step [33300/41412], Loss: 2.7190, Perplexity: 15.1649
Epoch [1/1], Step [33400/41412], Loss: 2.1694, Perplexity: 8.75294
Epoch [1/1], Step [33500/41412], Loss: 2.4596, Perplexity: 11.7006
Epoch [1/1], Step [33600/41412], Loss: 2.2454, Perplexity: 9.44447
Epoch [1/1], Step [33700/41412], Loss: 2.6903, Perplexity: 14.7357
Epoch [1/1], Step [33800/41412], Loss: 2.1667, Perplexity: 8.72958
Epoch [1/1], Step [33900/41412], Loss: 2.1093, Perplexity: 8.24289
Epoch [1/1], Step [34000/41412], Loss: 1.8862, Perplexity: 6.59415
Epoch [1/1], Step [34100/41412], Loss: 1.8389, Perplexity: 6.28943
Epoch [1/1], Step [34200/41412], Loss: 2.6413, Perplexity: 14.03157
Epoch [1/1], Step [34300/41412], Loss: 2.8755, Perplexity: 17.7344
Epoch [1/1], Step [34400/41412], Loss: 2.5730, Perplexity: 13.1047
Epoch [1/1], Step [34500/41412], Loss: 2.1192, Perplexity: 8.32448
Epoch [1/1], Step [34600/41412], Loss: 2.1835, Perplexity: 8.87742
Epoch [1/1], Step [34700/41412], Loss: 2.2763, Perplexity: 9.74070
Epoch [1/1], Step [34800/41412], Loss: 2.2215, Perplexity: 9.22087
Epoch [1/1], Step [34900/41412], Loss: 2.1565, Perplexity: 8.64075
Epoch [1/1], Step [35000/41412], Loss: 2.6285, Perplexity: 13.8529
Epoch [1/1], Step [35100/41412], Loss: 2.3457, Perplexity: 10.4410
Epoch [1/1], Step [35200/41412], Loss: 2.5379, Perplexity: 12.6532
Epoch [1/1], Step [35300/41412], Loss: 2.0472, Perplexity: 7.74613
Epoch [1/1], Step [35400/41412], Loss: 2.4071, Perplexity: 11.1016
Epoch [1/1], Step [35500/41412], Loss: 2.4892, Perplexity: 12.0521
Epoch [1/1], Step [35600/41412], Loss: 2.6532, Perplexity: 14.1988
Epoch [1/1], Step [35700/41412], Loss: 2.1744, Perplexity: 8.79736
Epoch [1/1], Step [35800/41412], Loss: 2.1367, Perplexity: 8.47135
Epoch [1/1], Step [35900/41412], Loss: 2.1382, Perplexity: 8.48422
Epoch [1/1], Step [36000/41412], Loss: 2.1638, Perplexity: 8.70413
Epoch [1/1], Step [36100/41412], Loss: 3.1768, Perplexity: 23.9691
Epoch [1/1], Step [36200/41412], Loss: 2.3585, Perplexity: 10.5755
```

```
Epoch [1/1], Step [36300/41412], Loss: 2.3094, Perplexity: 10.0679
Epoch [1/1], Step [36400/41412], Loss: 2.3484, Perplexity: 10.4692
Epoch [1/1], Step [36500/41412], Loss: 2.9291, Perplexity: 18.7109
Epoch [1/1], Step [36600/41412], Loss: 2.3930, Perplexity: 10.9465
Epoch [1/1], Step [36700/41412], Loss: 2.2514, Perplexity: 9.501004
Epoch [1/1], Step [36800/41412], Loss: 2.2222, Perplexity: 9.22778
Epoch [1/1], Step [36900/41412], Loss: 2.3793, Perplexity: 10.7969
Epoch [1/1], Step [37000/41412], Loss: 2.0724, Perplexity: 7.94419
Epoch [1/1], Step [37100/41412], Loss: 1.8279, Perplexity: 6.22102
Epoch [1/1], Step [37200/41412], Loss: 2.1200, Perplexity: 8.33100
Epoch [1/1], Step [37300/41412], Loss: 2.0510, Perplexity: 7.77587
Epoch [1/1], Step [37400/41412], Loss: 2.6932, Perplexity: 14.7782
Epoch [1/1], Step [37500/41412], Loss: 2.2774, Perplexity: 9.75172
Epoch [1/1], Step [37600/41412], Loss: 1.7951, Perplexity: 6.02032
Epoch [1/1], Step [37700/41412], Loss: 2.6250, Perplexity: 13.8045
Epoch [1/1], Step [37800/41412], Loss: 2.3901, Perplexity: 10.9145
Epoch [1/1], Step [37900/41412], Loss: 3.3238, Perplexity: 27.7652
Epoch [1/1], Step [38000/41412], Loss: 2.1026, Perplexity: 8.18766
Epoch [1/1], Step [38100/41412], Loss: 1.8858, Perplexity: 6.59139
Epoch [1/1], Step [38200/41412], Loss: 2.7941, Perplexity: 16.3479
Epoch [1/1], Step [38300/41412], Loss: 2.0791, Perplexity: 7.99753
Epoch [1/1], Step [38400/41412], Loss: 2.2257, Perplexity: 9.26032
Epoch [1/1], Step [38500/41412], Loss: 2.2347, Perplexity: 9.34337
Epoch [1/1], Step [38600/41412], Loss: 2.0948, Perplexity: 8.12356
Epoch [1/1], Step [38700/41412], Loss: 2.4457, Perplexity: 11.5382
Epoch [1/1], Step [38800/41412], Loss: 2.7545, Perplexity: 15.7133
Epoch [1/1], Step [38900/41412], Loss: 2.0831, Perplexity: 8.02969
Epoch [1/1], Step [39000/41412], Loss: 2.5780, Perplexity: 13.1704
Epoch [1/1], Step [39100/41412], Loss: 2.3252, Perplexity: 10.2288
Epoch [1/1], Step [39200/41412], Loss: 2.3804, Perplexity: 10.8091
Epoch [1/1], Step [39300/41412], Loss: 1.9687, Perplexity: 7.16171
Epoch [1/1], Step [39400/41412], Loss: 1.9454, Perplexity: 6.99631
Epoch [1/1], Step [39500/41412], Loss: 2.0586, Perplexity: 7.83526
Epoch [1/1], Step [39600/41412], Loss: 2.2452, Perplexity: 9.44216
Epoch [1/1], Step [39700/41412], Loss: 2.1534, Perplexity: 8.61416
Epoch [1/1], Step [39800/41412], Loss: 2.3863, Perplexity: 10.8728
Epoch [1/1], Step [39900/41412], Loss: 2.2877, Perplexity: 9.85195
Epoch [1/1], Step [40000/41412], Loss: 2.3080, Perplexity: 10.0542
Epoch [1/1], Step [40100/41412], Loss: 1.9571, Perplexity: 7.07877
Epoch [1/1], Step [40200/41412], Loss: 2.3574, Perplexity: 10.5632
Epoch [1/1], Step [40300/41412], Loss: 1.8887, Perplexity: 6.610608
Epoch [1/1], Step [40400/41412], Loss: 2.2880, Perplexity: 9.85522
Epoch [1/1], Step [40500/41412], Loss: 2.2742, Perplexity: 9.72063
Epoch [1/1], Step [40600/41412], Loss: 2.3768, Perplexity: 10.7706
Epoch [1/1], Step [40700/41412], Loss: 2.6832, Perplexity: 14.6312
Epoch [1/1], Step [40800/41412], Loss: 3.4377, Perplexity: 31.1146
Epoch [1/1], Step [40900/41412], Loss: 2.4492, Perplexity: 11.5791
Epoch [1/1], Step [41000/41412], Loss: 2.1068, Perplexity: 8.22208
```

```
Epoch [1/1], Step [41100/41412], Loss: 1.9256, Perplexity: 6.85937
Epoch [1/1], Step [41200/41412], Loss: 2.2202, Perplexity: 9.20969
Epoch [1/1], Step [41300/41412], Loss: 2.1618, Perplexity: 8.68659
Epoch [1/1], Step [41400/41412], Loss: 2.7445, Perplexity: 15.5573
Epoch [1/1], Step [41412/41412], Loss: 1.9121, Perplexity: 6.76727
```

Step 3: (Optional) Validate your Model

To assess potential overfitting, one approach is to assess performance on a validation set. If you decide to do this **optional** task, you are required to first complete all of the steps in the next notebook in the sequence (**3_Inference.ipynb**); as part of that notebook, you will write and test code (specifically, the `sample` method in the `DecoderRNN` class) that uses your RNN decoder to generate captions. That code will prove incredibly useful here.

If you decide to validate your model, please do not edit the data loader in **data_loader.py**. Instead, create a new file named **data_loader_val.py** containing the code for obtaining the data loader for the validation data. You can access: - the validation images at filepath `'/opt/cocoapi/images/train2014/'`, and - the validation image caption annotation file at filepath `'/opt/cocoapi/annotations/captions_val2014.json'`.

The suggested approach to validating your model involves creating a json file such as this one containing your model's predicted captions for the validation images. Then, you can write your own script or use one that you find online to calculate the BLEU score of your model. You can read more about the BLEU score, along with other evaluation metrics (such as TEOR and Cider) in section 4.1 of this paper. For more information about how to use the annotation file, check out the website for the COCO dataset.

[14]:
```
!nvidia-smi
```

```
Sat Oct 19 19:10:40 2024
+-----------------------------------------------------------------------------
----------+
| NVIDIA-SMI 560.35.03              Driver Version: 560.35.03      CUDA Version:
12.6     |
|-----------------------------------------+------------------------+-----------
----------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile
Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util
Compute M. |
|                                         |                        |
MIG M. |
|=========================================+========================+===========
==========|
|   0  NVIDIA L4                      Off |   00000000:36:00.0 Off |
0 |
| N/A   47C    P0             27W /   72W |    1166MiB /  23034MiB |      0%
Default |
|                                         |                        |
N/A |
```

```
+---------------------------------------+----------------------+-----------
----------+

+-----------------------------------------------------------------------------
----------+
| Processes:
|
|  GPU   GI   CI        PID   Type   Process name
GPU Memory |
|        ID   ID
Usage      |
|=============================================================================
==========|
|    0   N/A  N/A     23664      C   …conda3/envs/deepLearning/bin/python
476MiB |
|    0   N/A  N/A     28758      C   …conda3/envs/deepLearning/bin/python
676MiB |
+-----------------------------------------------------------------------------
----------+
```

[ ]: