Homework - 2

CPSC8430: Deep Learning

Due Date: March-25-2021

Naman Kulshreshtha

Sequence To Sequence Model (S2VT):

Training:

We have used a sequence-to-sequence RNN model for video captioning. To preprocess the video data, we create a 4096 long feature tensor containing data for 80 frames of the video that are provided as a numpy array. We also create a vocabulary from the training caption data provided.
To resolve the issue of vanishing gradients, we use GRUs instead of LTSMs, where one acts as an encoder and another as a decoder. Before passing the input to the GRU, we pass it through an embedding layer of size 512. The input to the encoder is a tensor padded with 0 to fill the empty spaces. The output of the GRU containing hidden state data from all the time steps is saved for usage in attention.

Steps to Improve Accuracy:

Generally, in S2VT models without attention, the hidden state at the last time is used as a context and fed into the decoder. The issue with this approach is that we are only feeding data from the last step and losing all the context from the previous steps. The attention layer helps the decoder focus on relevant parts of the input by scoring the relationship between each input word[1].

- Many attention mechanisms are present, ranging from general to dot product. In the out model, we are using the dot product one. We input all the hidden state data of the encoder to the attention layer, with another hidden state(h) received from the decoder when <BOS> is inputted to the decoder. These parameters are fed into the attention layer, and we receive the new weights.
- We also use teacher forcing (schedule sampling), where we train the decoder with the actual caption as input instead of the output of the previous timestamp. Using teacher forcing teaches the decoder to learn from the actual caption so that the decoder will have better output.

A context vector is first created by batch multiplying the attention weight and the decoder output. This context vector is concated with the tensor created from passing the captions through another embedding layer to create the input to the decoder.
The decoder's output is then passed through a linear layer and a softmax layer. The softmax output is used to get the word from the vocab. Repeat this for the maximum sentence size, after which the decoding process is complete.
To test the model and prevent overfitting, we use beam search with a beam size of 3.

Model Training Parameters:
Maximum Sentence Size:22.
Minimum size of word to be included in vocab: 2.

[1] https://zhanghanduo.github.io/post/attention/

Training dataset size:1400.
Validation dataset size:49.
Learning Rate:1e-3.
Optimizer: Adam.
Hidden state size:512.
Batch Size:64.
Feature_size:4096(already given).
Seq_len:80(length of the input feature NumPy array).
Epochs:50.

Model Performance:

BLEU@1 **without beam search** is **0.7166232345955359**
BLEU@1 with beam search and with a beam size of **3** is: **0.7208266636769638**