Assignment 4

ReadMe

Part 1 :

We analyzed the dataset and found that for each image, five captions were provided. So we created the mappings for each caption to the image for training the models right.

We use the ***load_image()*** function for loading the images and resizing them to 244*244 shape. Then we create the batches of size 32 for feeding to the model. Using Keras preprocessing, we preprocess the captions, and also give the <start> and <end> tags.

The final dataset is created, which contains an image and its caption's embedding mapping. Firstly we extract the features from the VGG16 model and then pass them to the encoder.

The output of the encoder is given to the decoder. For the predictions, the hidden state of the decoder is passed to the model.

For the evaluation of the predicted captions, we use the BLEU metric, BLEU-1, BLEU-2, BLEU-3, and BLEU-4. BLEU metric is used for matching predictions to the actual captions. BLEU-1, BLEU-2, BLEU-3, and BLEU-4 calculate the individual Cumulative N-gram score by assigning weights. For example for BLEU-1, weights are (1,0,0,0) for BLEU-2 weights are (0.5,0.5,0,0) and so on.

METEOR, which evaluates on a word to word basis assigns the score between 0 to 1, is also used for the evaluation of captions.

Helper Functions:

1. mapping_fun() : For loading the image array based from image name.
2. plot_attention() : For plotting the attention using attention weights, for evaluation.

3. Clean_Tagging() :For tagging the captions with <start> and <end>
4. convert_to_DataFrame() : For storing the cleaned captions with image names in the DataFrame.
5. pad_sequences() : For padding the data
6. train_step() : For training the model
7. train_data(): For training on the encoder decoder architecture
8. BLEU(): For BLEU 1-4 scores.
9. METEOR(): For METEOR score.

Part 2:

The given dataset contains two XML files, one for Laptop and one for Restaurant reviews. These XML files were parsed using ***xml.etree.ElementTree*** library.

Two dataframe were created for each of the XML files containing Context, Target, and Polarity for the sentence. Those sentences were dropped, which do not have any review that is 'Polarity' in the aspect term. Only those entries are considered, which contains the Polarity as positive, negative, or neutral.

After the data is preprocessed, we create embeddings using the GloVe embedding model. For target and context, two different embeddings are made as they would be fed to the IAN model.

We also tried ***oversampling*** to further enhance the accuracy of attention models, on restaurant data, but there is no increase in the accuracy.

Helper Functions:

1. makeDF(): Extract data from .xml files.
2. max_length_sentence(): To check max length sentences in the data.
3. simplify(): To preprocess input.
4. preprocessing(): Create embeddings & vectors for the input data.

5. plotting_epochs()
6. WithouAttention(): Class for the model, where no attention is applied.
7. IAN(): Class of interactive attention model
8. run(): To run the models described in 6 & 7 points.
9. evaluate(): For getting accuracies and class wise accuracies.
10. getting_attention_weights()
11. CharVal(): For attention weights plotting.

Contributions:

Question 1: Palak Tiwari (MT20103), Deepankar Kansal.

Question 2: Deepankar Kansal(MT20007), Mohd. Naki (2018052), Palak Tiwari.