Assignment 3

Deadline: Friday, 12th June (23:59)

29th May, 2020

Question 1 – Aspect-level Sentiment Classification (10pt)

Build a aspect-level classification model based on document-level and aspect-level data as proposed in:

> R. He, WS. Lee, HT. Ng, D. Dahlmeier, *Exploiting document knowledge for aspect-level sentiment classification*, 2018 (https://arxiv.org/abs/1806.04346).

Build an attention-based aspect-level sentiment classification model with Bidirectional Long Short Term Memory networks (BiLSTM). Your model shall include:

- BiLSTM network that learns sentence representations from input sequences (Recommend to use Bidirectional provided by Keras to define the BiLSTM network).
- Attention network that predicts sentiment label, given the representation weighted by the attention score.
- Fully connected network that predicts sentiment label, given the representation weighted by the attention score.

Requirements:

- You shall train your model based on transferring learning. That is, you need first train your doc-level model on document-level examples. Then the learned weights will be used to initialize aspect-level model and fine tune it on aspect-level examples.
- You shall use the alignment score function in attention network as same as the recommended paper. $f_{score}(h,t) = tanh(h^T W_a t)$.
- You shall evaluate trained model on the provided test set and show the accuracy on test set.

Data Description:

Document-level and aspect-level data sets are the same as practice-5.1.2 and can be download in:`https://surfdrive.surf.nl/files/index.php/s/AytwhaLUbIGRsCt`. The raw data set contains two domains: (1) Restaurant reviews; and (2) Electronics reviews. But please use lt_14 as experimental data. You can use the preprocessing notebook in practice-5.1.2 to process raw data.

Question 2 – Image Caption Generation (10pt)

Construct a Long-Short-Term-Memory (LSTM) network which takes an image representation obtained from a convolutional neural network (ConvNet) as input, and produces a caption describing the image. This task is based on:

> Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan, *Show and Tell: A Neural Image Caption Generator*, CVPR, 2015. `https://arxiv.org/abs/1411.4555`

You can use the Jupyter notebook `2IMM10_Assignment_3_1.ipynb`, which already downloads and loads the data, and provides some helper functions. You can write your code between all two consecutive occurrences of “`# ...`”. See the text cells in the notebook for additional information.

Data: Flickr8k

The *Flickr8k* dataset contains 8091 RGB images and 5 human-provided textual descriptions for each image (captions). For this task, the dataset has already been preprocessed:[1]

- All images have been rescaled to 128×128 RGB.
- Punctuation and special tokens have been removed from the captions.
- Words which occur less than 5 times in the whole corpus have been removed.
- All words have been converted to lower case.

Task 2.1: Generate Neural Codes (1pt)

Generate ConvNet representations (neural codes) for all images in the Flickr8k dataset. To this end, use the last convolutional layer ('*Conv_1*') of *MobileNetV2* pretrained on *Imagenet*.[2] This layer contains $4 \times 4 \times 1280$ features, yielding codes of length 20480.

Task 2.2: Analyze Captions (2pt)

Retrieve some information from the captions. In particular:

- Find and report the *maximal caption length*.
- Construct a collection of all words occurring in the captions and count their occurrences. Report the 10 most frequent words. Do you note a bias in the dataset?
- Include the special word '_' (the stop word, signaling the end of the captions) in the collection of words.
- How many unique words are there in the corpus, including '_'?
- Construct a mapping (dictionary) from words to integers as follows:
  - Stop word '_' $\rightarrow$ 0
  - Most frequent word $\rightarrow$ 1
  - Second most frequent word $\rightarrow$ 2
  - ...
- Construct an inverse mapping (dictionary), which maps integers back to words.

Task 2.3: Train Model (3pt)

Implement the model from the paper. In particular:

- Embed both the image codes and each word in a 512 dimensional space.
  - For the image codes use a fully connected layer, mapping the codes of length 20480 to 512 features. This layer should be subject to training.
  - Embed the integer encoded words using an *Embedding* layer (which is essentially a lookup table) of length 512. This layer should also be subject to training.
- Use the image and caption embeddings as inputs to an LSTM as discussed in the paper. Use 500 units for the LSTM.

[1] The Jupyter notebook automatically downloads the data from `https://surfdrive.surf.nl/files/index.php/s/kOIDM5tQPzv6IID`. **Please don't distribute.**

[2] The pretrained MobileNetV2 can conveniently be downloaded within Keras.

- Use a fully connected layer with *softmax* activation mapping the output of the LSTM to a distribution over words (in their integer encoding).
- How does the input and output need to be organized? For how many time steps T should the LSTM be unrolled? For each time step, $t = 0, \ldots, T-1$, which embedding should be input to the LSTM and what should be the target?

Train the model by minimizing *crossentropy*.

- Use Adam with a learning rate 0.001.
- Learn for maximal 100 epochs. Use early stopping with *patience* 1, providing the separate validation set.
- Use dropout with rate 0.5 for the LSTM.
- Evaluate and report the final training and validation loss.
- **Hint:** Use the sparse version of the crossentropy loss, in order to avoid memory issues.

Task 2.4: Generate Test Captions (4pt)

Implement a greedy decoder model as described in the paper ("beam search with a beam size of 1"). The decoder is akin to the trained model from Task 1.3. However, rather than providing image codes *and* captions, the decoder takes only the image codes as input.

- Equip the decoder with the weights from the trained model.
- Use the decoder to predict captions for all test images.
- Show 10 random test images and their predicted captions. Categorize the predictions as in Figure 5 in the paper.
- Compute and report the BLEU-1, BLEU-2, BLEU-3, and BLEU-4 scores over the test set.
- **Hint:** Use the *nltk* package to compute the BLEU scores.

Question 3 – Peer review (0pt)

Finally, each group member must write a single paragraph outlining their opinion on the work distribution within the group. Did every group member contribute equally? Did you split up tasks in a fair manner, or jointly worked through the exercises? Do you think that some members of your group deserve a different grade from others?