ADVANCED LEARNING FOR TEXT AND GRAPH DATA

Lab session 1: Hierarchical Attention Network

Lecture: Prof. Michalis Vazirgiannis
Lab: Dr. Guokan Shang and Yang Zhang

Tuesday, October 8, 2024

This handout includes theoretical introductions, coding tasks and questions. Before the deadline, you should submit on moodle **or** here a **.zip** file named `Lab<x>_lastname_firstname.pdf` containing a `/code/` folder (itself containing your scripts with the gaps filled) and an answer sheet, following the template available here, and containing your answers to the questions. Your answers should be well constructed and well justified. They should not repeat the question or generalities in the handout. When relevant, you are welcome to include figures, equations and tables derived from your own computations, theoretical proofs or qualitative explanations. **One submission is required for each student. The deadline for this lab is October 15, 2024 11:59 PM**. No extension will be granted. Late policy is as follows: $]0, 24]$ hours late $\rightarrow$ -5 pts; $]24, 48]$ hours late $\rightarrow$ -10 pts; > 48 hours late $\rightarrow$ not graded (zero).

---

1 Notebook

The notebook handout is available on Google Colab via this link:
https://colab.research.google.com/drive/10gMu-fk93FV2Ef982iGTSalryIrl83OT?usp=sharing
Make sure to execute a GPU runtime in case you are using cuda.

2 Learning objective

In this lab, you will get familiar with recurrent neural networks (RNNs), self-attention, and the HAN architecture [11]. In this architecture, sentence embeddings are first individually produced, and a document embedding is then computed from the sentence embeddings.
Requirements: we will use Python 3.x and **PyTorch 1.12.x**.
Dataset and reprocessing: In this lab, we will classify movie reviews (positive/negative) from the Internet Movie Database (IMDB) dataset. We have already preprocessed the raw data, and turned each review into an array of integers of shape `(1,doc_size,sent_size)`, where `doc_size` is the maximum number of sentences allowed per document and `sent_size` the maximum number of words allowed per sentence. Shorter sentences were padded with the special padding token, while shorter documents were padded with entire sentences containing `sent_size` times the special padding token. Longer documents and sentences were truncated. The integers correspond to indexes in a vocabulary that has been constructed from the training set, and in which the most frequent word has index 2. 0 and 1 are respectively reserved for the special padding token and the out-of-vocabulary token.

3 Model

3.1 RNN with GRU units

By definition, a RNN takes as input a sequence of vectors $(x_1, \ldots, x_T)$ of length T and returns a sequence of hidden representations $(h_1, \ldots, h_T)$, as shown in Fig. 1. The hidden representations are often called *annotations* in the literature. The hidden representation at time t, h_t, is computed from the current input x_t and the previous hidden representation h_{t-1}.

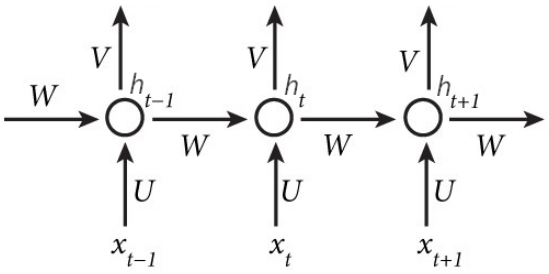

Figure 1: 3 steps of an unrolled RNN.

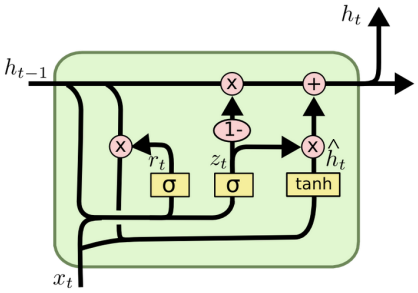

Figure 2: GRU unit. Taken from *Chris Olah's blog*.

As shown in Fig. 2, the GRU unit [2] is a RNN unit with two gates:

$$\text{reset gate: } r_t = \sigma\big(U_r x_t + W_r h_{t-1} + b_r\big) \tag{1}$$

$$\text{update gate: } z_t = \sigma\big(U_z x_t + W_z h_{t-1} + b_z\big) \tag{2}$$

Where σ denotes the sigmoid function. The candidate hidden state is computed as:

$$\hat{h}_t = \tanh\big(U_h x_t + W_h(r_t \circ h_{t-1}) + b_h\big) \tag{3}$$

The reset gate determines how much of the information from the previous time steps (stored in h_{t-1}) should be discarded. The new hidden state is finally obtained by linearly interpolating between the previous hidden state and the candidate one:

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \hat{h}_t \tag{4}$$

3.2 Self-attention

The attention mechanism [1] was developed in the context of encoder-decoder architectures for Neural Machine Translation (NMT) [8, 3], and rapidly applied to related tasks such as image captioning [10] and summarization [7]. Attention has also been used in encoder-only settings [11, 5]. In that case, it is called *self* or *inner* attention. Self-attention is also a core component of the recently introduced *transformer* architecture [9][1], which is at the basis of BERT [4]. Transformer and BERT have been game changers in NLP.

The basic idea behind attention is that rather than considering the last annotation h_T as a summary of the entire sequence, which is prone to information loss, the annotations at *all* time steps are used. The self-attention mechanism computes a weighted sum of the annotations, where the weights are determined by trainable parameters. Eq. 5 illustrates self-attention in its most simple variant. The annotations (h vectors) are first passed to a dense layer. The alignment coefficients (stored in the α

[1] note that transformer does not make use of RNNs!

vector) are then computed by comparing the output of the dense layer with a trainable context vector u and normalizing with a softmax. The attentional vector s is finally obtained as a weighted sum of the annotations.

$$u_t = \tanh(Wh_t) \qquad \alpha_t = \frac{\exp(u_t^\top u)}{\sum_{t'=1}^{T} \exp(u_{t'}^\top u)} \qquad s = \sum_{t=1}^{T} \alpha_t h_t \tag{5}$$

The context vector can be interpreted as a representation of the optimal element, on average (elements can be characters, subwords, words, sentences...). When faced with a new example, the model uses this knowledge to decide how much it should pay attention to each element in the sequence. The context vector is initialized randomly and updated during training.

Task 1
Fill the gap in the `AttentionWithContext` class.

Question 1 (5 points)
How can the basic self-attention mechanism be improved? You may read [5] to find ideas.

Question 2 (5 points)
Read the transformer paper [9]. What are the main motivations for replacing recurrent operations with self-attention?

3.3 Hierarchical self-attention

An interesting application of self-attention was provided by [11] and is illustrated in Fig. 3. This architecture is called HAN (for Hierarchical Attention Network). At each level, a RNN topped by a self-attention mechanism is used. First, the sentence embeddings are obtained. Then, a vector for the full document is computed from the sentence embeddings.

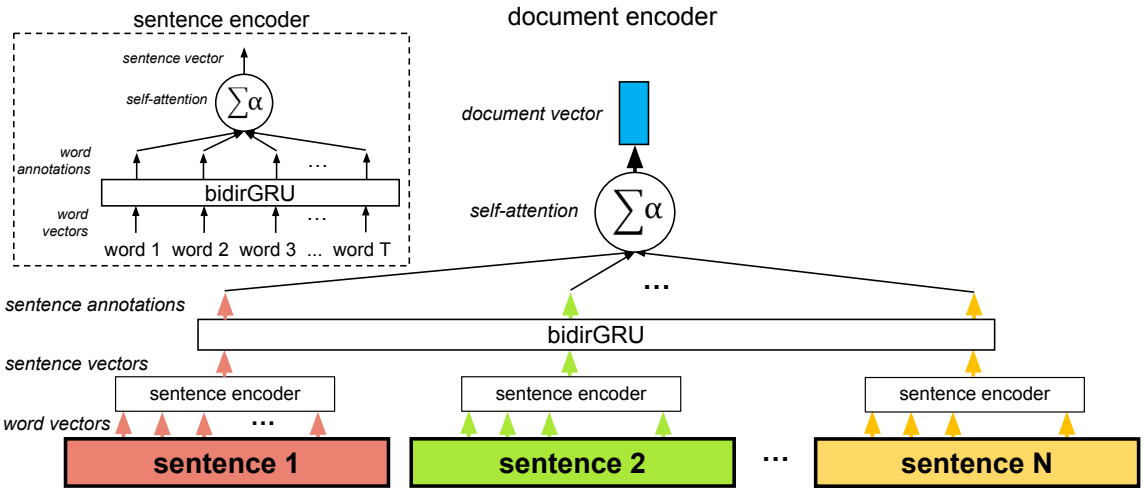

Figure 3: Hierarchical attention network.

HAN makes sense for two reasons: first, it matches the natural hierarchical structure of documents (words → sentences → document). Second, in computing the encoding of the document, it allows the model to first determine which words are important in each sentence, and then, which sentences are important overall. By being able to re-weigh the word attentional coefficients by the sentence attentional coefficients, the model captures the fact that a given instance of a word may be very important

when found in a given sentence, but another instance of the same word may not be as important when found in another sentence.

Task 2
Fill the gaps in sections "Data Loading", "Defining Architecture" and "Training" inside the Notebook and train your model.

Task 3
Fill the gaps in the last three sections of the notebook.

Question 3 (5 points)
Paste (or even better, plot) your attention coefficients for a document of your choice. Interpret your results.

Question 4 (5 points)
What are some limitations of the HAN architecture? You may read [6] to find ideas.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[3] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[6] Jean-Baptiste Remy, Antoine Jean-Pierre Tixier, and Michalis Vazirgiannis. Bidirectional context-aware hierarchical attention network for document understanding. *arXiv preprint arXiv:1908.06006*, 2019.

[7] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

[8] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[10] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[11] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489, 2016.