Natural Language Processing **Tel Aviv University**

Assignment 2: Language Models

Due Date: *May 20, 2025* Lecturer: Dr. Mor Geva, TA: Idan Tarshish

1 Word-Level Neural Bi-gram Language Model

In this question, you will implement and train neural language model, and evaluate it on using perplexity.

(a) Derive the gradient with respect to the input of a softmax function when cross entropy loss is used for evaluation, i.e., find the gradients with respect to the softmax input vector $\boldsymbol{\theta}$, when the prediction is made by $\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{\theta})$. Cross entropy and softmax are defined as:

$$\text{CE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_i y_i \cdot \log(\hat{y}_i)$$

$$\text{softmax}(\boldsymbol{\theta})_i = \frac{\exp(\theta_i)}{\sum_j \exp(\theta_j)}$$

The gold vector $\boldsymbol{y}$ is a one-hot vector, and the predicted vector $\hat{\boldsymbol{y}}$ is a probability distribution over the output space.

(b) Derive the gradients with respect to the input $\boldsymbol{x}$ in a one-hidden-layer neural network (i.e., find $\frac{\partial J}{\partial \boldsymbol{x}}$, where J is the cross entropy loss $\text{CE}(\boldsymbol{y}, \hat{\boldsymbol{y}})$). The neural network employs a sigmoid activation function for the hidden layer, and a softmax for the output layer. Assume a one-hot label vector $\boldsymbol{y}$ is used. The network is defined as:

$$\boldsymbol{h} = \sigma(\boldsymbol{x}\boldsymbol{W}_1 + \boldsymbol{b}_1),$$
$$\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{h}\boldsymbol{W}_2 + \boldsymbol{b}_2).$$

The dimensions of the vectors and matrices are $\boldsymbol{x} \in \mathbb{R}^{1\times D_x}, \boldsymbol{h} \in \mathbb{R}^{1\times D_h}, \hat{\boldsymbol{y}} \in \mathbb{R}^{1\times D_y}, \boldsymbol{y} \in \mathbb{R}^{1\times D_y}$. The dimensions of the parameters are $\boldsymbol{W}_1 \in \mathbb{R}^{D_x\times D_h}, \boldsymbol{W}_2 \in \mathbb{R}^{D_h\times D_y}, \boldsymbol{b}_1 \in \mathbb{R}^{1\times D_h}, \boldsymbol{b}_2 \in \mathbb{R}^{1\times D_y}$.

(c) Implement the forward and backward passes for a neural network with one sigmoid hidden layer. Fill in your implementation in `q1c_neural.py`. Sanity check your implementation with `python q1c_neural.py`.

(d) GloVe (Global Vectors) embeddings are a type of word embeddings that represent words as vectors in a high-dimensional space, based on the co-occurrence statistics of words in a corpus. They are related to the skip-gram embeddings you saw in class in that they both aim to capture the semantic and syntactic relationships between words, but GloVe embeddings incorporate global corpus-level information in addition to local context information. In this section you will be using GloVe embeddings to represent the vocabulary. Use the neural network to implement a bigram language model in `q1d_neural_lm.py`. Use GloVe embeddings to represent the vocabulary (`data/lm/vocab.embeddings.glove.txt`). Implement the `lm_wrapper` function, that is used by `sgd` to sample the gradient, and the `eval_neural_lm` function that is used for model evaluation. Report the dev perplexity in your written solution. Don't forget to include `saved_params_40000.npy` in your submission zip!

2 Generating Shakespeare with a Character-level Language Model

In this section, we will train a language model and use it to generate text. Follow the instructions, complete the code, and answer the questions in the attached notebook `q2_char_rnn_generation.ipynb`. Feel free to comment inside the notebook.

3 Perplexity

(a) Show that perplexity calculated using the natural logarithm $\ln(x)$ is equal to perplexity calculated using $\log_2(x)$. i.e:

$$2^{-\frac{1}{M}\sum_{i=1}^{M}\log_2 p(s_i|s_1,\ldots,s_{i-1})} = e^{-\frac{1}{M}\sum_{i=1}^{M}\ln p(s_i|s_1,\ldots,s_{i-1})}$$

(b) In this section you will be computing the perplexity of your previous trained models on two different passages. Please provide your results in the PDF file, as well as attach the code to your code files (You can simply add the new code to what you wrote in the previous sections, along with a descriptive comment). The two different passages appear in the `.zip` file you have got. Their names are: `shakespeare_for_perplexity.txt` which contains a subset from the Shakespeare dataset, and `wikipedia_for_perplexity.txt` which contains a certain passage from Wikipedia. For your convenience, both passages are also provided in a POS-tagged format, similar to the datasets used for training and testing in Section 1.

Please compute the perplexity of the bi-gram LM (from Section 1) and the character-level LM (from Section 2) on both these passages.

(c) Try to explain the results you have got. Particularly, why there might be large gaps in perplexity, while looking at different passages.

4 Implementing a Transformer Model

In this question we will implement a full functioning transformer, using toy data and building the architecture from the building blocks. **Open the attached notebook (`q4_transformer.ipynb`) and fill in the missing code parts according to the instructions.**

After completing and running the notebook, answer these theoretical questions in your submitted pdf file:

1. Observe the assertion of

$$assert\ d\_model\ \%\ num\_heads == 0$$

   at the beginning of the '`__init__`' function in the Multiple Head Attention Class (Part 1 of the notebook) . Why is this assertion needed?

2. As mentioned throughout the notebook, our implemented Transformer is autoregressive. What changes would be required to adapt it into a Masked Language Model (MLM)?

   (a) Which components should be added or removed from the architecture?

   (b) How would the masking strategy differ?

5 Sentiment Analysis (practical)

In this exercise, you will be asked to browse solutions for specific downstream tasks in NLP. The goal of this exercise is to help you become comfortable with using other researchers' code and common tools. We hope that this practical experience will aid you in your final project. We encourage you to continue diving deeper with critical thinking and creative ideas!

Papers with code is free and open resource platform with Machine Learning papers, code, datasets, methods and evaluation tables (by Meta AI Research). You can browse state-of-the-art solutions for various tasks according to categories, get familiar with thousands of benchmark datasets, compare models using different evaluation metrics, and much more.

Explore the sentiment analysis on text challenge. Go to the task lead-board to see whether the article code is available in the description table. Currently, the best model (in the leadboard) for this task is T5-11B (We can see it's not the most active research topic, but it's nevertheless a good way to introduce you to some practical tools! Also, note that you will not load T5-11B in this exercise but T5-Small, due to limited resources).

Include the code you used for this section in the Colab Notebook, and answer to the questions in the pdf.

1. Download the article "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer" (top accuracy on this task). It's OK if you don't fully understand the meaning of this article and its solution! You will learn more about it during the course. Find the link to their GitHub repository and copy it here. Once you've opened their GitHub, answer the following questions: Which model have they made publicly available? Which dataset is used to benchmark sentiment analysis? How do you evaluate success in this task?

2. In the article, T5 was pre-trained on multiple datasets and fine-tuned for specific downstream tasks. The original repository only published the weights for the pre-trained version (and not for the fine-tuned version used in the sentiment analysis task).

   Search for a T5 model fine-tuned on the SST2 dataset on the HuggingFace platform, and use it as demonstrated in the Hugging face platform tutorial.

   LLMs (I bet most of you are familiar with the term; you will learn more about it very soon) come in different sizes. Use the **T5-Small** pre-trained version.

   Please state exactly which model will you use and load.

   Now, use the model to predict the following sentences (print each result in a different cell in the Colab Notebook):

   - "This movie is awesome"
   - "I didn't like the movie so much"
   - "I'm not sure what I think about this movie."
   - "Did you like the movie?"

3. Load the SST2 dataset and evaluate the accuracy of this model on it. Does it match the results declared on Papers with Code?

4. Is the data in the SST2 dataset balanced? Why is this an important question when evaluating the model's performance with respect to this dataset?

5. Can you think on some properties of sentiment analysis evaluation that human-evaluators may notice but are not considered when evaluating the accuracy of this dataset??

6. (Bonus:) Imagine you are consulting with friends who work in a healthcare organization that has recently approved the use of an alternative supplier for one of their drugs. They want to monitor client satisfaction with both suppliers, ensuring they are equally satisfactory to patients. To achieve this, they have collected a dataset consisting of summaries from visits to a family doctor. They have confirmed that the dataset includes patients treated with each of the two drug suppliers. Now, they would like to classify the visit summaries as positive or negative to analyze the results. Since this dataset is quite unique, they seek your advice on how to approach this task.

   Please suggest some general high-level approaches to solve the task. Explain to them which factors they should take into account and what the advantages and disadvantages are of the different approaches.