CS 4782 Homework 2

Due: 3/03/26 11:59pm on Gradescope

Late submissions accepted until 3/05/26 11:59 pm

Problem 1: Word2Vec (25 points)

1. **Understanding Skip-Gram Model**

   The key insight behind the Skip-Gram model is that *"a word is known by its neighbor words"*, as we expect similar words to appear together. In Skip-Gram prediction model, we will have a center word t, and a contextual window surrounding t as $t + j$. For example, in Figure 1, we have the center word "banking", and a contextual window with the words "turning", "into", "crises", and "as".

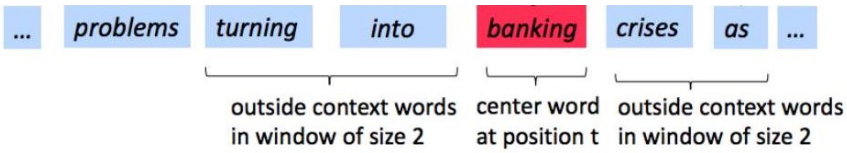

Figure 1: An example of Skip-Gram model with window size 2. Here we want to maximize the probability $P(\text{"turning"} \mid \text{"banking"})$, $P(\text{"into"} \mid \text{"banking"})$, $P(\text{"crises"} \mid \text{"banking"})$, and $P(\text{"as"} \mid \text{"banking"})$.

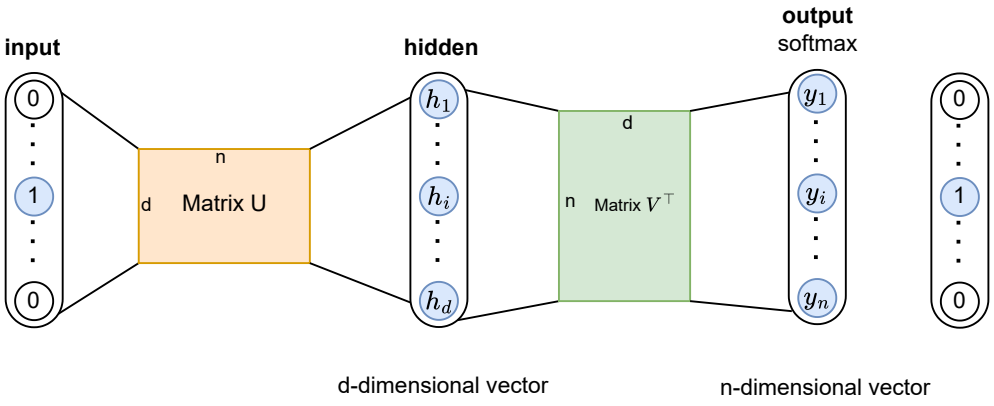

Figure 2: The Skip-Gram architecture as shown in the lecture slides. Matrices $\mathbf{U} \in \mathbb{R}^{d \times n}$ and $\mathbf{V} \in \mathbb{R}^{d \times n}$ represent the learnable vectors for the center word vectors and contextual vectors, respectively. Both $\mathbf{U}$ and $\mathbf{V}$ contain a vector for every word in the vocabulary set $\mathcal{V}$. d and n represent the dimension of the hidden word and number of words in the vocabulary set $\mathcal{V}$, respectively. Assume you have little data, so you decide to share the weights across the two layers, i.e. $\mathbf{U} = \mathbf{V}$.

The goal of the Skip-Gram algorithm is to accurately learn the probability distribution $P(x_{t+j} \mid x_t)$,

where x_t is the center word and x_{t+j} is a word inside the contextual window. We want to maximize the conditional probability as a softmax probability:

$$P(x_{t+j} \mid x_t) = \frac{\exp\left(\mathbf{U}_{:,t}{}^{\top}\mathbf{V}_{:,t+j}\right)}{\sum_{v\in\mathcal{V}}\exp\left(\mathbf{U}_{:,t}{}^{\top}\mathbf{V}_{:,v}\right)} \tag{1}$$

The above equation is retrieving specific vectors from the weight matrices $\mathbf{U}$ and $\mathbf{V}$. As a reminder, $\mathbf{U} \in \mathbb{R}^{d\times n}$, so $\mathbf{U}_{:,t}$ is retrieving the learnable vector associated with the t^{th} center word, whereas $\mathbf{V}_{:,t+j}$ is retrieving the learnable vector associated with the $t+j^{th}$ contextual word ($\mathbf{V} \in \mathbb{R}^{d\times n}$).

And the Skip-Gram "naive softmax loss" will have the following form

$$\mathcal{L} = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\neq 0}\log P(x_{t+j} \mid x_t) \tag{2}$$

$$= -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\neq 0}\left(\mathbf{U}_{:,t}{}^{\top}\mathbf{V}_{:,t+j} - \log\sum_{v\in\mathcal{V}}\left(\exp\left(\mathbf{U}_{:,t}{}^{\top}\mathbf{V}_{:,v}\right)\right)\right) \tag{3}$$

Here, m represents the contextual window size.

Suppose we have a vocabulary of size 5 $\{x_a, x_b, x_c, x_d, x_e\}$, and we have a $\mathbf{U} = \mathbf{V} = \begin{bmatrix} 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & 0 \\ 1 & 1 & -1 & 1 & -1 \end{bmatrix}$.

We are given a sequence of text as $[x_a, x_e, x_c, x_b, x_d, x_c]$

(a) Suppose we have a center word x_c as the 3rd position in the given text, and we have a window size of 2. Please identify the words inside the contextual window. (2 points)

(b) Please compute the $P(x_i|x_c)$ for all words x_i inside the contextual window of sizes 2 with the word x_c as the center word. (5 points)

(c) What are the limitations of using the naive softmax loss shown in Equation 3 in a large text dataset in terms of computational efficiency? Please justify your answer. (5 points)

(d) **Negative sampling** is commonly used as an alternative to the naive softmax loss. The idea of negative sampling loss is that for each (context word, center word) pair, we will create a random subset $\mathcal{V}_S = \{x_1, \ldots, x_k\} \subset \mathcal{V}$ of size k without replacement, and each word in $\mathcal{V}_S$ **is not** inside the contextual window of the center words.

We can then change the problem of maximizing the conditional probability in Equation 1 into a **binary classification problem** between predicting whether a certain word is a contextual word. In this case, the probability of an "*is contextual word*" relationship between a word x_{t+j} and x_t is defined as $\sigma(\mathbf{U}_{:,t}{}^{\top}\mathbf{V}_{:,t+j})$ where σ is the sigmoid function.

In particular, we want to find $\mathbf{U}$ and $\mathbf{V}$ to jointly maximize the probability of a correct contextual word and minimize the probability of a false/negative contextual word as

$$P(x_{t+j}|x_t)\prod_{i=1}^{k}(1 - P(x_i|x_t)) \tag{4}$$

Then, we will have negative sampling loss as

$$\mathcal{L}_{\text{NS}} = -\left(\prod_{t=1}^{T}\prod_{-m\le j\le m, j\neq 0}\left(P(x_{t+j}|x_t)\prod_{k=1}^{K}(1 - P(x_k|x_t))\right)\right)^{1/T} \tag{5}$$

i. Please simplify $-\log(-\mathcal{L}_{\text{NS}})$ for the Skip-Gram model using Equation 5. (Hint: $1-\sigma(x) = \sigma(-x)$)(7 points)

ii. The idea of negative sampling is prevalent in unsupervised learning in general. Here please identify at least 2 advantages of negative sampling loss over the naive softmax loss in Skip-Gram model. (Hint: let's think about the computational efficiency / separation in the embedding space by negative sampling). (6 points)

Problem 2: Exploring the Mathematics Behind Attention (25 points)

1. We saw in class that $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{\text{QK}^\intercal}{\sqrt{\text{d}_\text{k}}}\right)\text{V}$. Let $A = QK^\intercal$. In terms of K's key vectors and Q's query vectors (both of which are row vectors), what is $A_{i,j}$? (3 points)
2. Assume that you have random vectors $x, y \sim \mathcal{N}(\mu, \sigma^2 I)$ with $x, y, \mu \in \mathbb{R}^d$, $\sigma > 0$, and $I \in \mathbb{R}^{d\times d}$. What is $E[x^\intercal y]$? (4 points)
3. Let $\mu = 0$ and $\sigma = 1$. What is $\text{Var}[\text{x}^\intercal\text{y}]$? (Hint: You might find some of these properties useful) (10 points)
4. Assuming that the key and query vectors in K and Q similarly have 0 mean and covariance matrix I. Using your answers to previous parts, what is $\text{Var}[\text{A}_{\text{i,j}}]$? (2 points)
5. In the formula for attention, what effect does dividing by $\sqrt{d_k}$ have on the variance of each element of $\frac{QK^\intercal}{\sqrt{d_k}}$? Why might this be useful? (6 points)

Problem 3: Transformers vs RNNs (15 points)

Compare and contrast transformers with RNNs for the following aspects:

1. Briefly explain why vanishing/exploding gradients appear in RNNs during optimization. (2 points)
2. RNNs aren't great at capturing long-term dependencies; explain how transformers address these issues. (3 points)
3. Discuss the time complexity of both architectures with respect to sequence length and how they scale with longer sequences. (5 points)
4. Applications: Provide examples of tasks where transformers are particularly effective compared to RNNs, and vice versa. (5 points)

Problem 4: Read a Foundational Paper (16 points)

Read Attention is All You Need (2017), and answer the following questions. This paper is foundational for nearly all of deep learning, and the following questions will make sure that you understand the key methods and results of the paper. **Please answer all questions in three sentences or less.** The point of the questions is simply to check your comprehension.

1. What is the main motivation for using self-attention instead of recurrence or convolution? (4 points)
2. How does the transformer model take into account the relative positions of tokens? What do they mention as other possible ways to do so? (4 points)

3. Briefly compare the transformer model's performance and efficiency to other state of the art models. Cite a specific table or score in your answer. (As a note, the BLEU scoring method, or Bilingual Evaluation Understudy score, compares generated sentences to reference sentences using n-grams.) (4 points)

4. Which part of the paper interested you most? Briefly describe something you found exciting about the methods, results, or anything else. (4 points)

Problem 5: Self Attention by Hand (19 Points)

In this problem, we'll have you compute one pass through of single-head attention on a given matrix X, which contains 3-dimensional tokens representing words (for simplicity, one token corresponds to one word here).

We are going to be looking at the phrase, "dogs and cats are cute." Row X_1 represents the word "dogs," X_2 represents the word "and," and so on. (Note: For the simplicity of this problem, we will assume that this matrix already has positional embeddings included.)

$$X = \begin{bmatrix} 2 & 1 & 0 \\ 3 & -1 & 0 \\ 1 & 1 & 0 \\ -3 & -1 & 0 \\ 4 & 0 & 1 \end{bmatrix}$$

Let's assume that the weights of the key, query and value matrices are initialized randomly as below:

$$W_Q = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, W_K = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, W_V = \begin{bmatrix} 0.2 \\ 0.4 \\ 0.3 \end{bmatrix}$$

1. Fill in the values of matrix $\text{softmax}(\frac{\text{QK}^\text{T}}{\sqrt{\text{d}_\text{k}}})$ below, and show your work along the way. (8 points)

$\text{softmax}(\frac{\text{QK}^\text{T}}{\sqrt{\text{d}_\text{k}}}) =$

	dogs	and	cats	are	cute
dogs					
and					
cats					
are					
cute					

2. Write a sentence about something you notice in the table above. What do different rows and columns represent? What is "paying attention" to what? (3 points)

3. Now, find Attention(Q, K, V). (4 points)

4. Let's assume we are applying look-ahead masking to our matrix. (4 points)

   (a) What is the purpose of look-ahead masking, and what is the general structure of a look-ahead masking matrix?

   (b) Do we apply this masking *before* or *after* we do the softmax, and why?