CS 601.471/671 NLP: Self-supervised Models

Homework 4: Language Modeling + N-Grams + Fixed-Window LMs

For homework deadline, check the calendar on the course website*

Name: ____________________

Collaborators, if any: ____

Sources used for your homework, if any: ______________________________

This assignment focuses on language modeling, while continuing to build up on our prior knowledge of neural networks. We will review several aspects about training neural nets and also extend it to modeling sequences in language.

How to hand in your written work: Via Gradescope as before.

1 Concepts, intuitions and big picture

1. Suppose you have built a neural network. You decide to initialize the weights and biases to be zero. Which of the following statements are True? (Check all that apply)
   ☐ Each neuron in the first hidden layer will perform the same computation. So even after multiple iterations of gradient descent each neuron will be computing the same thing as other neurons in the same layer.
   ☐ Each neuron in the first hidden layer will perform the same computation in the first iteration. But after one iteration of gradient descent they will learn to compute different things because we have "broken symmetry".
   ☐ Each neuron in the first hidden layer will compute the same thing, but neurons in different layers will compute different things, thus we have accomplished "symmetry breaking" as described in lecture.
   ☐ The first hidden layer's neurons will perform different computations from each other even in the first iteration; their parameters will thus keep evolving in their own way.
   Answer: *TBD*

2. Vectorization allows you to compute forward propagation in an L-layer neural network without an explicit for-loop (or any other explicit iterative loop) over the layers $l = 1, 2, \times, L$. True/False?
   ☐ True
   ☐ False
   Answer: *TBD*

3. The `tanh` activation usually works better than sigmoid activation function for hidden units because the mean of its output is closer to zero, and so it centers the data better for the next layer. True/False?
   ☐ True
   ☐ False
   Answer: *TBD*

4. Which of the following techniques does NOT prevent a model from overfitting?
   ☐ Data augmentation ☐ Dropout ☐ Early stopping ☐ None of the above
   Answer: *TBD*

5. Why should dropout be applied during training? Why should dropout NOT be applied during evaluation?
   Answer: *TBD*

6. Explain why initializing the parameters of a neural net with a constant is a bad idea.
   Answer: *TBD*

*`https://self-supervised.cs.jhu.edu/sp2023/`

7. You design a fully connected neural network architecture where all activations are sigmoids. You initialize the weights with large positive numbers. Is this a good idea? Explain your answer.
   Answer: *TBD*

8. What are some benefits to using Fixed-Window-LM over trigrams (or n-grams generally speaking?)
   Answer: *TBD*

9. Explain what is the importance of "residual connections".
   Answer: *TBD*

10. The n-gram models we've shown make the n-th order Markov assumption, i.e. that the distribution of words depends only on the previous $n-1$ words. What properties of language will it not capture? Discuss (briefly) several distinct ways in which this assumption is false for natural language.
    Answer: *TBD*

11. Follow-up to previous question: Despite the Markov assumption, n-gram models are remarkably good at predicting the next word. Discuss why this might be. What information is in the previous word(s) that makes these models perform so surprisingly well? In particular, what kinds of grammatical information do they capture?
    Answer: *TBD*

12. Explain how are perplexity and cross-entropy loss related?
    Answer: *TBD*

13. For a vocabulary of $|V|$ words, what would you expect perplexity to be if your model predictions were completely random? Compute the corresponding cross-entropy loss for $|V| = 2000$ and $|V| = 10000$, and keep this in mind as a baseline.
    Answer: *TBD*

2 Dead[ly] neurons

The ReLU activation function can lead to "dead neurons" which may never be activated on any input, i.e. their output is zero for any input. Let's consider a two-layer feedforward network with N input nodes, H nodes in the hidden layer with matrix weights $W^{(1)}$, and bias $b^{(1)}$ and a scalar output y with weights $W^{(2)}$. We can formulate the layers of our network as follows:

$$\begin{cases} \mathbf{h} = \text{ReLU}(W^{(1)}\mathbf{x} + b^{(1)}) \\ \hat{y} = W^{(2)}.\mathbf{h} \end{cases}$$

Note, you can rewrite the first layer computation for each dimension $i \in \{0, ..., H\}$ as following: $h_i = \text{ReLU}(W_i^{(1)}.\mathbf{x} + b_i^{(1)}) = \text{ReLU}(\sum_{j=1}^{N} W_{ij}^{(1)} x_j + b_i^{(1)})$, where $W_i^{(1)}$ is the i-th slice of the parameter matrix $W^{(1)}$ and $W_{ij}^{(1)}$ is the (i,j)-th scalar element of the parameter matrix. We optimize the parameters of our model to minimize an arbitrary, differentiable loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ which takes as arguments the model predictions and the ground truth labels.

1. Under what conditions is node h_i dead? Your answer should be expressed in terms of the parameters $W_i^{(1)}$ and $b_i^{(1)}$. **Hint:** As we defined earlier, "dead neurons" are never activated on any input, i.e. their output is zero, for any input. Specifically, ReLU's are never activated is their input is always negative.
   Answer: *TBD*

2. Derive the gradients $\frac{\partial \ell}{\partial W_{ij}^{(1)}}$ and $\frac{\partial \ell}{\partial b_i^{(1)}}$ using the chain rule. Assume that the partial derivative of the loss on a given instance is $\frac{\partial \ell}{\partial y} = c$.
   Answer: *TBD*

3. Using your answers to the previous two parts, explain why a "dead" neuron can never be "brought back to life". That is, if a neuron is firing zero for all the inputs in training data, its parameters will never change with gradient updates and hence. As a result, it will eternally remain "dead"! :(
   Answer: *TBD*

3 Softmax Smackdown: Squishing the Competition

3.1 Softmax temperature

Remember the softmax function, $\sigma(\mathbf{z})$? Here we will add a *temperature* parameter $\tau \in \mathbb{R}^+$ to this function:

$$\text{Softmax: } \sigma(\mathbf{z};\tau)_i = \frac{e^{z_i/\tau}}{\sum_{j=1}^{K} e^{z_j/\tau}} \quad \text{for } i = 1, \dots, K$$

Show the following:

1. In the limit as temperature goes to zero $\tau \to 0$, softmax becomes the same as greedy action selection, `argmax`.
   Answer: *TBD*

2. In the limit as temperature goes to infinity $\tau \to +\infty$, softmax gives equiprobable selection among all actions.
   Answer: *TBD*

3.2 The cousin function: `log-sum-exp`

Let $x_1, ..., x_n$ be real values. Let,

$$f(x_1, ..., x_n) = \log \sum_{i=1}^{n} \exp(x_i).$$

This is called `log-sum-exp` function. Note that this function is a scalar function $\mathbb{R}^n \to \mathbb{R}$ and it is different from the "softmax" transformation which convert any score vector to a probability vector.

1. Calculate the gradient of f w.r.t. vector $\mathbf{x} = (x_1, ..., x_n)^\top$.
   Answer: *TBD*

2. Prove that:

$$\max_i x_i \leq f(x_1, ..., x_n) \leq \max_i (x_i) + \log n$$

   Answer: *TBD*

Aside: The naming of "softmax" is a misnomer. In reality, what it does it does is more like `soft-argmax` function. In contrast, "softmax" would have been a better name for `log-sum-exp` function.

4 n-gram Language Models

4.1 A toy n-gram language model

Consider the following vocabulary, $V = \{\texttt{BOS}, \texttt{EOS}, \texttt{here}, \texttt{Adam}, \texttt{are}, \texttt{you}\}$ where BOS is the dummy token indicating the beginning of a sentence, and EOS indicates the end of a sentence. Consider the following training data:

```
BOS here you are EOS
BOS here you are Adam EOS
BOS are you here EOS
BOS you are here EOS
BOS you are here EOS
BOS Adam you are here EOS
BOS you are EOS
```

1. Compute all n-gram counts up to $n = 2$.
   Answer: *TBD*

2. Calculate the following probabilities: $\text{P}(\texttt{you}), \text{P}(\texttt{you}|\texttt{are})$.
   Answer: *TBD*

3. Using unigram and bigram language models, compute the probabilities of the following sentences:
   $S_1 = \texttt{BOS here you are EOS}$
   $S_2 = \texttt{BOS here you are Adam EOS}$
   $S_3 = \texttt{BOS Adam here you are Adam EOS}$
   $S_4 = \texttt{BOS here you are here you are EOS}$
   Answer: *TBD*

Words, they come and they go
Building sentences, with a flow
But what if we could predict the next
With a language model, at its best

Enter N-gram, the model so grand
Breaking down words, hand in hand
With each n, a new understanding
Of the language, expanding

And then came Neural, with power untold
A model, that learned, as words unfold
No longer just counting, it knows the why
A model, that's smarter, it never says die

With layers deep, it learns to perceive
The patterns and structure, we so believe
Predicting the next, with such grace
A model, that's changing, the world's embrace

So let us celebrate, N-gram and Neural
In the world of language models, so real
For they're tools, that help us to see
The patterns and structure, so vividly.

–ChatGPT Feb 9 2023

4.2 Expected number of "unseen" bigrams

Consider a simple language model in which each token is drawn independently from the vocabulary V with uniform probability $1/|V|$, independent of all other tokens.

1. Given a corpus of size M that is generated this way, what is the expectation on the fraction of all possible bigrams that have zero counts? For simplicity of derivations assume that $|V|$ is large enough that $\frac{1}{|V|} \approx \frac{1}{|V|-1}$.
   Answer: *TBD*

2. **Extra Credit:** For a $\varepsilon \in (0, 1)$, determine the range of values of M such that the fraction of bigrams with zero count is at most ε. For simplicity of derivations, you can use the following approximation: $\ln(1 + x) \approx x$ for $x \approx 0$.
   Answer: *TBD*

3. In natural languages (such as English, ...), word probabilities are not uniform, i.e., $p(w) \neq \frac{1}{|V|}$. In this case, prove that the expected fraction of unseen bigrams will be higher than the one computed for (1).
   Answer: *TBD*

4.3 Extra Credit: Validity of the n-gram distribution

The goal of this homework is to prove that n-gram language models give valid probabilities if the n-gram probabilities are valid. Specifically, we will show that n-gram LMs provide a proper probability distribution over:

- (A) sequences of *finite* length, without an end-of-sequence symbol EOS;
- (B) all sequences, including those of *infinite* length, if provided with an end-of-sequence symbol EOS.

Setup. Assume that we are given n-gram probabilities $\text{P}(w_t|w_{t-1}, \ldots, w_{t-n+1})$ for a fixed n. And assume that $\sum_{w_t \in V} \text{P}(w_t|w_{t-1}, \ldots, w_{t-n+1}) = 1$ that is computed over all words w_t in the vocabulary V, using contexts $(w_{t-1}, \ldots, w_{t-n+1})$.

1. Fix ℓ such than $\ell \geq 1$. Then, let $V^\ell \subset (V \backslash \texttt{EOS})^*$ denote the set of all sequences of length ℓ that do not include the stop symbol.[*] Prove that $\sum_{\mathbf{x} \in V^\ell} \mathrm{P}(\mathbf{x}) = 1$, if $\mathrm{P}(\mathbf{x})$ is the joint probability of all words in the sequence $\mathbf{x} \in V^\ell$. Your proof should proceed by induction. You should handle the start-of-string case where the context is $n-1$ start symbols $\mathbf{P}(w_1 | \underbrace{\texttt{BOS}, \cdots, \texttt{BOS}}_{n-1})$, but do not include the stop symbol.

   Answer: *TBD*

2. If we do not include the stop symbol, n-gram models define a valid probability distribution over sequences of a *fixed length*, as shown above. As a consequence, they do not define a valid probability distribution over V^+ (c.f., footnote [*]). Show that by including the stop symbol $\texttt{EOS}$, n-gram models define a valid distribution over sequences of all lengths. That is, show $\sum_{\mathbf{x} \in V^+} \mathrm{P}(\mathbf{x}) = 1$, if $\mathrm{P}(\mathbf{x})$ is the probability of the sequence of words.
   Answer: *TBD*

5 Programming

See the course website for the link to Google Colab: Colab Link

6 Optional Feedback

Have feedback for this assignment? Found something confusing? We'd love to hear from you!

[*] In this problem, $.^*$ and $.^+$ refer to Kleene star and plus: `https://en.wikipedia.org/wiki/Kleene_star`.