Natural Language Understanding, Generation, and Machine Translation (2021–22)

School of Informatics, University of Edinburgh
Frank Keller

Tutorial 2: Neural Network Language Models (Week 4)

This tutorial includes both calculation questions and more open-ended questions. Question 3 is intended to help you think about how to apply models you've seen before to new problems. *Most* of the points on the exam will come from questions of this form, that require you to think through a new, open-ended scenario.

1 The Softmax Function

The softmax function takes an arbitrary vector $\mathbf{v}$ as input, with $|\mathbf{v}|$ dimensions. It computes an output vector, also of $|\mathbf{v}|$ dimensions, whose ith element is given by:

$$\text{softmax}(\mathbf{v})_i = \frac{\exp(\mathbf{v}_i)}{\sum_{j=1}^{|\mathbf{v}|} \exp(\mathbf{v}_j)}$$

Question 1: Softmax Function

a. What is the purpose of the softmax function? the activation function in the output layer of neural network models that predict a multinomial probability distribution.

b. What is the purpose of the expression in the numerator? map v to exp(v), so v can choose from -infinite to infinite and the result is always positive

c. What is the purpose of the expression in the denominator? guarantee the sum of P(v_i) is 1

Now consider how a neural language model with a softmax output layer compares with a classic n-gram language model. Typically, we use techniques like smoothing or backoff in conjunction with n-gram models.

d. Does this problem arise in the neural model? Why or why not? No. Even v_i = 0, exp(v_i)=1, which solves unknown problem

2 Feedforward Language Models

Consider a **feedforward language model** of the type discussed in lecture 5. In this model, the probability $P(w_i \mid w_{i-n+1}, ..., w_{i-1})$ is given by:

$$\begin{aligned}
P(w_i \mid w_{i-n+1}, ..., w_{i-1}) &= \text{softmax}(\mathbf{V}\mathbf{h}_2 + \mathbf{b}_2) \\
\mathbf{h}_2 &= \tanh(\mathbf{W}\mathbf{h}_1 + \mathbf{b}_1) \\
\mathbf{h}_1 &= \text{concatenate}(\mathbf{C}\mathbf{w}_{i-n+1}, \ldots, \mathbf{C}\mathbf{w}_{i-1}) \\
\mathbf{w}_i &= \text{onehot}(w_i) \qquad \triangleleft \text{ for all } i
\end{aligned}$$

In this notation, $\mathbf{V}$, $\mathbf{W}$, and $\mathbf{C}$ are matrices, while $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{h}_1$, $\mathbf{h}_2$, and $\mathbf{w}_{i-n+1}, \ldots, \mathbf{w}_{i-1}$ are vectors. The parameters of the model are $\mathbf{V}$, $\mathbf{W}$, $\mathbf{C}$, $\mathbf{b}_1$, and $\mathbf{b}_2$, while the remaining variables are intermediate layers computed by the network.

Now consider the number of parameters required to represent this model. This number is determined by the size of the vocabulary (given to you by the data), the order n, and the dimension of the two hidden layers, $\mathbf{h}_1$ and $\mathbf{h}_2$, which we will denote d_1 and d_2, respectively (Note that the first dimension must be divisible by n, but you can ignore this detail in your calculations). Dimensions d_1 and d_2 are modeling choices, though the practical consideration is how they impact the model's accuracy.

Question 2: Parameters of Neural Nets

a. How would you express the number of model parameters in terms of $|V|$, n, d_1, and d_2? |V|*n*d_1+d_1*d_2+d_1+d_2 (weight + bias)

b. An effective size for the hidden dimension of a neural NLP model is often in the hundreds. For n from 2 to 5, how many parameters would your model have if $d_1 = d_2 = 100$? What if $d_1 = d_2 = 1000$?

c. What do you conclude about the relative memory efficiency of classic n-gram and feedforward neural language models? If you increased n even further, what would happen? feedforward is better, because it uses all the words in the input, while n-gram model denpends on n. If n grows bigger, N-gram need exponential computation resourses, feedforward just grow linearly.

d. How would you expect the number of parameters in an RNN model to scale with $|V|$? 2*|V|, one for output, one for previous cell

e. **[Advanced, for now]** Can you think of any strategies to substantially reduce the number of parameters?

The next problem looks at how to apply neural models you've just learned about to a problem you've seen in previous courses: part-of-speech tagging. Given an input sentence $x = x_1 \dots x_{|x|}$, we want to predict the corresponding tag sequence $y = y_1 \dots y_{|x|}$. Let x_i denote the ith word of x, y_i denote the ith word of y, and $|x|$ denote the length of x. Note that $|y| = |x|$. For example:

$i =$	1	2	3	4	5	6	7	8	9	10
$x_i =$	Each	day	starts	with	one	or	two	lectures	by	researchers
$y_i =$	DT	NN	VBZ	IN	CD	CC	JJR	NNS	IN	NNS

We have access to many training examples like this, and our goal is to model the conditional probability of the tag sequence given the sentence, that is: $P(y \mid x)$. There are many possible choices here. To simplify the problem, let's *assume* that each element of y is conditionally independent of each other. That is, we want to model:

$$P(y \mid x) = \prod_{i=1}^{|y|} P(y_i \mid x)$$

P(y_i|x) = softmax(Vh_2 + b_2)

input: whole sentence

output: the probability of part of speech tag

Question 3: Model Design

a. Design a feedforward neural network to model $P(y_i \mid x)$. Identify any independence assumptions you make. Draw a diagram that illustrates how the model computes probabilities for the tag of the word "with": What is the input, and how is the output distribution computed from the input? Write out the basic equations of the model, and explain your choices.

b. Design an RNN to model $P(y_i \mid x)$. Identify any independence assumptions you make. Draw a diagram that illustrates how the model computes probabilities for the tag of the word "with": What is the input, and how is the output distribution computed from the input? Write out the basic equations of the model, and explain your choices.

c. **[Advanced, for now]** Can you model $P(y_i \mid x)$ without independence assumptions, using multiple RNNs?

For each question, the goal is to design a *simple* model for the distribution. You solution should only use architectures that we discussed in the first two weeks of the course. If you are aware of other architectures, you should not use them here.