Assignment 3

COMP7607: Natural Language Processing - University of Hong Kong

Fall 2022

Question 1: A long short-term memory (LSTM) is defined as follows. At time t it receives an input vector $\boldsymbol{x_t} \in \mathbb{R}^k$ of observations, an input vector $\mathbf{h}_{t-1} \in \mathbb{R}^k$ representing the previous hidden state, and a memory state $\mathbf{c}_{t-1} \in \mathbb{R}^k$ from the previous time step. The computes three gates $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ controlling, respectively. It additionally computes a new value for the memory $\mathbf{c}.t$ and a new hidden representation as follows:

$$
\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{I}_x\mathbf{x}_t + \mathbf{I}_h\mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{F}_x\mathbf{x}_t + \mathbf{F}_h\mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{O}_x\mathbf{x}_t + \mathbf{O}_h\mathbf{h}_{t-1} + \mathbf{b}_o) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot g(\mathbf{C}_x\mathbf{x}_t + \mathbf{C}_h\mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{h}_t &= \mathbf{o}_t \odot g(\mathbf{c}_t)
\end{aligned}
$$

where σ is the element-wise logistic sigmoid function and g is an element-wise nonlinearity (e.g., tanh). The behavior of the network is controlled by the parameters $\mathbf{I}_x$, $\mathbf{I}_h$, $\mathbf{F}_x$, $\mathbf{F}_h$, $\mathbf{O}_x$, $\mathbf{O}_h$, $\mathbf{C}_x$, and $\mathbf{C}_h$ which are all in $\mathbb{R}^{k\times k}$. The base values $\mathbf{h}_0 = \mathbf{c}_0 = \mathbf{0}$. Finally, a new output is computed:

$$\mathbf{y}_t = f(\mathbf{W}\mathbf{h}_t + \mathbf{b})$$

Question 1a: Please briefly explain the functionality of the three gates $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$. How do they control the LSTM?

Question 1b: Please briefly explain why are vanishing or exploding gradients an issue for RNNs, and how LSTM addresses this problem.

Question 2: Let $\mathbf{Q} \in \mathbb{R}^{N \times d}$ denote a set of N query vectors, which attend to M key and value vectors, denoted by matrices $\mathbf{K} \in \mathbb{R}^{M \times d}$ and $\mathbf{V} \in \mathbb{R}^{M \times c}$ respectively. For a query vector at position n, the softmax attention function computes the following quantity:

$$\mathrm{Attn}\left(\mathbf{q}_n, \mathbf{K}, \mathbf{V}\right) = \sum_{m=1}^{M} \frac{\exp\left(\mathbf{q}_n^\top \mathbf{k}_m\right)}{\sum_{m'=1}^{M} \exp\left(\mathbf{q}_n^\top \mathbf{k}_{m'}\right)} \mathbf{v}_m^\top := \mathbf{V}^\top \mathrm{softmax}\left(\mathbf{K}\mathbf{q}_n\right)$$

which is an average of the set of value vectors $\mathbf{V}$ weighted by the normalized similarity between different queries and keys.

Question 2a: Please briefly explain what is the time and space complexity for the attention computation from query $\mathbf{Q}$ to $\mathbf{K}$, $\mathbf{V}$, using the big O notation.

Question 3: Consider context-free grammar with the following rules (assume that S is the start symbol):

$S \rightarrow NP\ VP$
$NP \rightarrow NP\ PP$
$PP \rightarrow IN\ NP$
$VP \rightarrow V\ NP$
$VP \rightarrow VP\ PP$
$NP \rightarrow$ we
$NP \rightarrow$ sushi
$NP \rightarrow$ chopsticks
$IN \rightarrow$ with
$V \rightarrow$ eat

Question 3a: How many parse trees are there under this grammar for the following sentence? Draw them.

we eat sushi with chopsticks

Question 4: Consider a probabilistic context-free grammar with the following rules (assume that S is the start symbol):

$\mathrm{S} \rightarrow \mathrm{NP\ VP}$	1.0
$\mathrm{VP} \rightarrow \mathrm{Vt\ NP}$	0.7
$\mathrm{VP} \rightarrow \mathrm{VP\ PP}$	0.3
$\mathrm{NP} \rightarrow \mathrm{DT\ NN}$	0.8
$\mathrm{NP} \rightarrow \mathrm{NP\ PP}$	0.2
$\mathrm{PP} \rightarrow \mathrm{IN\ NP}$	1.0
$\mathrm{Vi} \rightarrow$ sleeps	1.0
$\mathrm{Vt} \rightarrow$ saw	1.0
$\mathrm{NN} \rightarrow$ man	0.1
$\mathrm{NN} \rightarrow$ woman	0.1
$\mathrm{NN} \rightarrow$ telescope	0.3
$\mathrm{NN} \rightarrow$ dog	0.5
$\mathrm{DT} \rightarrow$ the	1.0
$\mathrm{IN} \rightarrow$ with	0.6
$\mathrm{IN} \rightarrow$ in	0.4

Question 4a: What's the most likely parse tree for the following sentence under this PCFG? Show the CYK chart you developed below.

the man saw the woman with the dog

Question 4b: What's the (marginal) probability of the following sentence under this PCFG?

the man saw the woman with the dog

Question 4c: For an input of length M and a grammar with R productions and N non-terminals, please briefly explain what is the time and space complexity of the CYK algorithm using big O notation.

Question 5: A trigram language model is also often referred to as a second-order Markov language model. It has the following form:

$$P\left(X_1 = x_1, \ldots, X_n = x_n\right) = \prod_{i=1}^{n} P\left(X_i = x_i \mid X_{i-2} = x_{i-2}, X_{i-1} = x_{i-1}\right)$$

Question 5a: Could you briefly explain the advantages and disadvantages of a high-order Markov language model compared to the second-order one?

Question 5b: Could you give some examples in English where English grammar suggests that the second-order Markov assumption is clearly violated?

Question 6: The goal of sequence labeling is to assign tags to words, or more generally, to assign discrete labels to discrete elements in a sequence. Given a sequence of n words $\boldsymbol{x}$, assign each a label from $\mathcal{L}$. Let $L = |\mathcal{L}|$. In this question, we will explore different approaches, all of which cast the problem as:

$$\hat{\boldsymbol{y}} = \underset{\boldsymbol{y} \in \mathcal{L}^n}{\operatorname{argmax}} \operatorname{Score}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})$$

	he	raises	purses
N	-2	-3	-2
V	-7	-2	-4

(a) Weights for emission features.

	N	V	⧫
◊	-1	-1	$-\infty$
N	-1	-1.5	-2
V	-2	-2.5	-0.5

(b) Weights for transition features. The "from"tags are on the columns, and the "to"tags are on the rows. Transition weight from START ◊ to STOP ⧫ is implicitly set to $-\infty$

Tabelle 1: Consider the minimal tagset { N, V } , corresponding to nouns and verbs. For the following question parts, assume that the log probabilities (log base 2) are as above.

Question 6a: Local classifier Define score of a word x_i getting label $y \in \mathcal{L}$ in context: $\operatorname{score}(\boldsymbol{x}, i, y; \boldsymbol{\theta})$, for example through a feature vector, $\mathbf{f}(\boldsymbol{x}, i, y)$. (Here, "$i$ " indicates the position of the input word to be classified.) Train a classifier to decode locally, i.e.,

$$\hat{y}_i = \underset{y \in \mathcal{L}}{\operatorname{argmax}} \operatorname{score}(\boldsymbol{x}, i, y; \boldsymbol{\theta})$$

The classifier is applied to each $x_1, x_2, \ldots$ in turn, but all the words can be made available at each position.

Question 6a1: Based on the provided features, what is the tag sequence for this sentence with this local classifier? Report the tag sequence and probability. (Do not need to calculate START ◊ and STOP ⧫ symbol;just tag sequence for the three words)

Question 6b: Sequential classifier Define score of a word x_i getting label y in context, including previous labels: $\text{score}\left(\boldsymbol{x}, i, \hat{\boldsymbol{y}}_{1:i-1}, y; \boldsymbol{\theta}\right)$. (From here, we won't always write $\boldsymbol{\theta}$, but the dependence remains.) Train a classifier, e.g.,

$$\hat{y}_i = \underset{y \in \mathcal{L}}{\text{argmax}}\, \text{score}\left(\boldsymbol{x}, i, \hat{\boldsymbol{y}}_{1:i-1}, y\right)$$

The classifier is applied to each $x_1, x_2, \ldots$ in turn. Each one depends on the outputs of preceding iterations.

Algorithm 1: Beam Search for Sequential Classifier

Data: $\boldsymbol{x}$ (length n), a sequential classifier's scoring function $score$, and beam width k
Result: Output: best-scored element of H_n
begin
  Let H_0 score hypotheses at position 0 , defining only $H_0(\langle\rangle) = 0$.;
  for $i \leftarrow 1$ **to** n **do**
    $C \longleftarrow \emptyset$;
    foreach *hypothesis* $\hat{\boldsymbol{y}}_{1:i-1}$ *scored by* H_{i-1} **do**
      foreach $y \in \mathcal{L}$ **do**
        place new hypothesis $\hat{\boldsymbol{y}}_{1:i}y \leftarrow H_{i-1}\left(\hat{\boldsymbol{y}}_{1:i}\right) + \text{score}\left(\boldsymbol{x}, i, \hat{\boldsymbol{y}}_{1:i-1}, y\right)$ into C.
      end
    end
    Let H_i be the k-best scored elements of C.
  end
end

Question 6b1: Greedy search What is the highest posterior probability tag sequence for this sentence with this sequential classifier? Report the tag sequence and probability. (including the ♦ transition)

Question 6b2: Beam search What is the tag sequence obtained by beam searching with a beam size of 2? Report the tag sequence and probability. (including the ♦ transition)

Question 6b3: For a sequence $\boldsymbol{x}$ in length n, label space $\mathcal{L}$, and beam width k, please explain the time and space complexity of the beam search algorithm.

Question 6c: Hidden Markov Model The Hidden Markov Model (HMM) approach should remind you of language models. The HMM is based on augmenting the Markov chain. A Markov chain makes a very strong assumption that if we want to predict the future in the sequence, all that matters is the current state. All the states before the current state have no impact on the future except via the current state.

$$\hat{y}_i = \underset{y \in \mathcal{L}}{\operatorname{argmax}} \operatorname{score}\left(\boldsymbol{x}, i, y_{i-1}, y\right)$$

Generally, we use the Viterbi algorithm for choosing $\hat{\boldsymbol{y}}$. It is an efficient algorithm using dynamic programming, an algorithmic technique for reusing work in recurrent computations. We begin by solving an auxiliary problem: rather than finding the best tag sequence, we compute the score of the best tag sequence.

Algorithm 2: The Viterbi algorithm.

Data: Each scores $s\left(\boldsymbol{x}, i, y, y'\right)$, for all $i \in \{0, \ldots, n\}, y, y' \in \mathcal{L}$
Result: $\hat{\boldsymbol{y}}$
begin
  create a matrix for Viterbi variables $v[L, n]$;
  foreach $y \in \mathcal{L}$ **do**
    $v[1, y] = s(\boldsymbol{x}, 0, \Diamond, y)$;
    $\mathrm{bp}[1, y] = 0$;
  end
  for $i \leftarrow 2$ **to** $n + 1$ **do**
    foreach $y \in \mathcal{L}$ **do**
      $v[i, y] = \max_{y_{i-1} \in \mathcal{L}} s(\boldsymbol{x}, i-1, y_{i-1}, y) + v[i-1, y_{i-1}]$;
      $\mathrm{bp}[i, y] = \underset{y_{i-1} \in \mathcal{L}}{\operatorname{argmax}} s(\boldsymbol{x}, i-1, y_{i-1}, y) + v[i-1, y_{i-1}]$;
    end
  end
  $\hat{y}_{n+1} \leftarrow \blacklozenge$;
  for $i \in \{n, \ldots, 1\}$ **do**
    $\hat{y}_i \leftarrow \mathrm{bp}_{i+1}\left(\hat{y}_{i+1}\right)$
  end
  return $\hat{\boldsymbol{y}}$;
end

Question 6c1: Fill in the Viterbi matrix v. Report the tag sequence and probability.

	he	raises	purses
N			
V			

Question 6c2: For a sequence $\boldsymbol{x}$ in length n, label space $\mathcal{L}$, please explain the time and space complexity of the Viterbi algorithm.

Question 7: Could you briefly compare and explain the differences between ELMo, BERT, GPT-2, and GPT-3 in different aspects such as architecture, parameters, ability, pretraining objectives, data, etc.?