Deep Learning for NLP 2020
Exercise 10 Solution

July 12, 2020

1 Pingo

Note: Sparse autoencoders are a variation where the dimension of the hidden layer is larger than the input and output dimension. A sparsity constraint enforces that only a limited number of neurons can be active (i.e. have a nonzero output) at a time.

- Which of the following statements about autoencoders are correct?
  - ☑ Autoencoders rely on self-supervision.
  - □ The major disadvantage of undercomplete autoencoders is their dimensionality reduction of the input data.
  - ☑ Sparse autoencoders enforce sparsity on the hidden layer.
  - □ Sparse autoencoders enforce sparsity on the output layer.
- Take a look at the following statements about encoder-decoder models. Which statements are true?
  - ☑ When using beam search in a decoder, the probability of an individual output value comes from a softmax layer.
  - □ To train an encoder-decoder model with attention, one needs training data with hard aligned sentence pairs.
  - ☑ Hard alignments do not have to be sparse.

2 End-to-End Models

Explain the difference between an end-to-end model and a pipeline model in up to two sentences. In addition, state an advantage and a disadvantage of end-to-end models.

Answer: End-to-end models convert raw input data into output data in one jointly trainable model. Pipeline models rely on a sequence of separately trained models (for example lemmatization, classification, any post-processing).

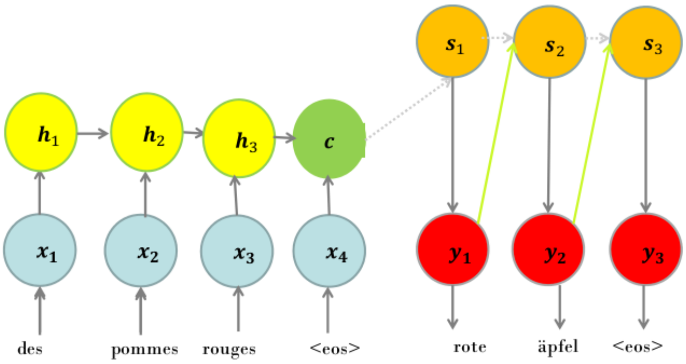
- advantage: does not suffer from error propagation as pipeline models (if lemmatizer only has accuracy of 80%, the pipeline output cannot be better than 80%).
- disadvantages: few or no knowledge about the input data (a neural machine translation model needs to learn translation, but also the syntax and grammar of all languages), hard to debug why an end-to-end model works (or why it doesn't)

3 Encoder-Decoder Model

In this task, you will use an encoder-decoder model to predict an output sequence for a given input sequence.

Encoder-Decoder Principle

We will use the basic formulation used by Sutskever et al. 2014[1]:



The prediction phase of such an encoder-decoder architecture has the following steps:

1. Feed each input vector $\mathbf{x}_t$ from the input sequence $\mathbf{x}$ to the encoder RNN. The outputs of the encoder RNN are irrelevant.
2. After feeding the end-of-sequence symbol, the hidden state of the encoder RNN represents the encoding of our input sequence $\mathbf{x}$. This value is referred to as context vector $\mathbf{c}$.
3. Set the hidden state of the decoder RNN to $\mathbf{c}$.
4. Decode the first output symbol by calculating $\mathbf{y}_1$ in the decoder RNN.
5. Decode all following output symbols by feeding $\mathbf{y}_{t-1}$ as an input, computing the hidden state, and observing $\mathbf{y}_t$. This step is repeated until $\mathbf{y}_t$ equals the end-of-sequence symbol.

RNN Formulation

To simplify this task, we will use the same vocabulary of symbols for encoding and decoding, and we will use the same RNN for encoding and decoding. We also omit any bias input to neurons.

We use the following RNN formulation (see lecture 08, slide 11):

$$\begin{aligned}\mathbf{h}_t &= \sigma_H(\mathbf{x}_t \cdot \mathbf{U} + \mathbf{h}_{t-1} \cdot \mathbf{W})\\ \mathbf{y}_t &= \sigma_Y(\mathbf{h}_t \cdot \mathbf{V})\end{aligned}$$

With:

- Input vectors $\mathbf{x}_t \in \mathbb{R}^{1\times n}$, where n is the input embedding dimensionality
- the RNN hidden state $\mathbf{h}_t \in \mathbb{R}^{1\times d}$, where d is the RNN hidden state size
- Output vectors $\mathbf{y}_t \in \mathbb{R}^{1\times m}$, where m is the size of the (output) vocabulary
- Hidden layer activation σ_H
- Output layer activation σ_Y
- Weight matrices $\mathbf{U} \in \mathbb{R}^{n\times d}$, $\mathbf{V} \in \mathbb{R}^{d\times m}$ and $\mathbf{W} \in \mathbb{R}^{d\times d}$

[1] https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

3.1 Encoding Phase

Assume $d = 2$, $n = 2$, $m = 3$, $\sigma_H = \tanh$ and $\sigma_Y = $ softmax. Additionally assume that training resulted in the following weight matrices:

$$\mathbf{U} = \begin{pmatrix} 0.60 & 2.42 \\ -1.92 & -2.42 \end{pmatrix}$$

$$\mathbf{V} = \begin{pmatrix} 0.65 & 0.20 & -0.16 \\ 0.77 & -1.36 & 0.70 \end{pmatrix}$$

$$\mathbf{W} = \begin{pmatrix} 1.05 & 0.27 \\ -0.97 & 1.05 \end{pmatrix}$$

Also

$$\mathbf{h}_0 = \begin{pmatrix} 0 & 0 \end{pmatrix}$$

Our vocabulary and the embeddings of each symbol within the vocabulary are defined as in table 1.

i	symbol	embedding
0	`<eos>`	$\begin{pmatrix} 0 & 0 \end{pmatrix}$
1	a	$\begin{pmatrix} 1.40 & 0.22 \end{pmatrix}$
2	b	$\begin{pmatrix} -0.15 & 0.12 \end{pmatrix}$

Table 1: Embeddings

Encode the input sequence **`a,a,<eos>`** and report the context vector $\mathbf{c}$. Two decimal places suffice for the calculation.

Solution:

$$\begin{aligned} \mathbf{h}_1 &= \sigma_H(\mathbf{x}_1 \cdot \mathbf{U} + \mathbf{h}_0 \cdot \mathbf{W}) \\ &= \sigma_H(\mathbf{x}_1 \cdot \mathbf{U}) \\ &= \tanh\left(\begin{pmatrix} 1.40 & 0.22 \end{pmatrix} \cdot \begin{pmatrix} 0.60 & 2.42 \\ -1.92 & -2.42 \end{pmatrix}\right) \\ &= \tanh\left(\begin{pmatrix} 0.42 & 2.86 \end{pmatrix}\right) \\ &= \begin{pmatrix} 0.40 & 0.99 \end{pmatrix} \end{aligned}$$

$$\begin{aligned} \mathbf{h}_2 &= \sigma_H(\mathbf{x}_2 \cdot \mathbf{U} + \mathbf{h}_1 \cdot \mathbf{W}) \\ &= \tanh\left(\mathbf{x}_2 \cdot \mathbf{U} + \begin{pmatrix} 0.40 & 0.99 \end{pmatrix} \cdot \begin{pmatrix} 1.05 & 0.27 \\ -0.97 & 1.05 \end{pmatrix}\right) \\ &= \tanh\left(\begin{pmatrix} 0.42 & 2.86 \end{pmatrix} + \begin{pmatrix} -0.54 & 1.15 \end{pmatrix}\right) \\ &= \tanh\left(\begin{pmatrix} -0.12 & 4.01 \end{pmatrix}\right) \\ &= \begin{pmatrix} -0.12 & 1.00 \end{pmatrix} \end{aligned}$$

$$\mathbf{h}_3 = \mathbf{c} = \sigma_H(\mathbf{x}_3 \cdot \mathbf{U} + \mathbf{h}_2 \cdot \mathbf{W})$$

$$
\begin{aligned}
&= \sigma_H(\mathbf{h}_2 \cdot \mathbf{W}) \\
&= \tanh\left(\begin{pmatrix}-0.12 & 1.00\end{pmatrix} \cdot \begin{pmatrix}1.05 & 0.27 \\ -0.97 & 1.05\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}-1.1 & 1.02\end{pmatrix}\right) \\
&= \begin{pmatrix}-0.8 & 0.76\end{pmatrix}
\end{aligned}
$$

3.2 Decoding Phase

Now, using $\mathbf{c} = \begin{pmatrix}-0.8 & 0.76\end{pmatrix}$, decode an output sequence until you encounter the **<eos>** symbol. Which sequence do you obtain?

Hint:

$$y_j = \text{softmax}(\mathbf{z})_j = \frac{\exp(\mathbf{z}_j)}{\sum_k \exp(\mathbf{z}_k)}$$

Solution:

$$
\begin{aligned}
\mathbf{s}_1 &= \mathbf{c} \\
\\
\mathbf{y}_1 &= \sigma_Y(\mathbf{s}_1 \cdot \mathbf{V}) \\
&= \text{softmax}\left(\begin{pmatrix}-0.8 & 0.76\end{pmatrix} \cdot \begin{pmatrix}0.65 & 0.20 & -0.16 \\ 0.77 & -1.36 & 0.70\end{pmatrix}\right) \\
&= \text{softmax}\left(\begin{pmatrix}0.07 & -1.19 & 0.66\end{pmatrix}\right) \\
&= \frac{1}{\exp(0.07) + \exp(-1.19) + \exp(0.66)} \cdot \begin{pmatrix}\exp(0.07) & \exp(-1.19) & \exp(0.66)\end{pmatrix} \\
&= \begin{pmatrix}0.32 & 0.09 & 0.58\end{pmatrix} \\
\mathbf{y}_{1,\text{decoded}} &:= \text{vocab}[\text{argmax}(\mathbf{y}_1)] = \text{b} \\
\\
\mathbf{s}_2 &= \sigma_H(\mathbf{y}_1 \cdot \mathbf{U} + \mathbf{s}_1 \cdot \mathbf{W}) \\
&= \tanh\left(\begin{pmatrix}-0.15 & 0.12\end{pmatrix} \cdot \begin{pmatrix}0.60 & 2.42 \\ -1.92 & -2.42\end{pmatrix} + \begin{pmatrix}-0.8 & 0.76\end{pmatrix} \cdot \begin{pmatrix}1.05 & 0.27 \\ -0.97 & 1.05\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}-0.32 & -0.65\end{pmatrix} + \begin{pmatrix}-1.58 & 0.58\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}-1.9 & -0.07\end{pmatrix}\right) \\
&= \begin{pmatrix}-0.96 & -0.07\end{pmatrix} \\
\\
\mathbf{y}_2 &= \sigma_Y(\mathbf{s}_2 \cdot \mathbf{V}) \\
&= \text{softmax}\left(\begin{pmatrix}-0.96 & -0.07\end{pmatrix} \cdot \begin{pmatrix}0.65 & 0.20 & -0.16 \\ 0.77 & -1.36 & 0.70\end{pmatrix}\right) \\
&= \text{softmax}\left(\begin{pmatrix}-0.68 & -0.1 & 0.1\end{pmatrix}\right) \\
&= \begin{pmatrix}0.2 & 0.36 & 0.44\end{pmatrix} \\
\mathbf{y}_{2,\text{decoded}} &:= \text{vocab}[\text{argmax}(\mathbf{y}_2)] = \text{b} \\
\\
\mathbf{s}_3 &= \sigma_H(\mathbf{y}_2 \cdot \mathbf{U} + \mathbf{s}_2 \cdot \mathbf{W})
\end{aligned}
$$

$$
\begin{aligned}
&= \tanh\left(\mathbf{y}_2 \cdot \mathbf{U} + \begin{pmatrix}-0.94 & -0.33\end{pmatrix} \cdot \begin{pmatrix}1.05 & 0.27 \\ -0.97 & 1.05\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}-0.32 & -0.65\end{pmatrix} + \begin{pmatrix}-0.94 & -0.33\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}-1.26 & -0.98\end{pmatrix}\right) \\
&= \begin{pmatrix}-0.85 & -0.75\end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{y}_3 &= \sigma_Y(\mathbf{s}_3 \cdot \mathbf{V}) \\
&= \text{softmax}\left(\begin{pmatrix}-0.85 & -0.75\end{pmatrix} \cdot \begin{pmatrix}0.65 & 0.20 & -0.16 \\ 0.77 & -1.36 & 0.70\end{pmatrix}\right) \\
&= \text{softmax}\left(\begin{pmatrix}-1.13 & 0.85 & -0.39\end{pmatrix}\right) \\
&= \begin{pmatrix}0.1 & 0.7 & 0.2\end{pmatrix} \\
\mathbf{y}_{3,\text{decoded}} &:= \text{vocab}[\text{argmax}(\mathbf{y}_3)] = \texttt{a}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{s}_4 &= \sigma_H(\mathbf{y}_3 \cdot \mathbf{U} + \mathbf{s}_3 \cdot \mathbf{W}) \\
&= \tanh\left(\mathbf{y}_3 \cdot \mathbf{U} + \begin{pmatrix}-0.85 & -0.75\end{pmatrix} \cdot \begin{pmatrix}1.05 & 0.27 \\ -0.97 & 1.05\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}0.42 & 2.86\end{pmatrix} + \begin{pmatrix}-0.17 & -1.02\end{pmatrix}\right) \\
&= \tanh\left(\begin{pmatrix}0.25 & 1.84\end{pmatrix}\right) \\
&= \begin{pmatrix}0.24 & 0.95\end{pmatrix}
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{y}_4 &= \sigma_Y(\mathbf{s}_4 \cdot \mathbf{V}) \\
&= \text{softmax}\left(\begin{pmatrix}0.24 & 0.95\end{pmatrix} \cdot \begin{pmatrix}0.65 & 0.20 & -0.16 \\ 0.77 & -1.36 & 0.70\end{pmatrix}\right) \\
&= \text{softmax}\left(\begin{pmatrix}0.89 & -1.24 & 0.63\end{pmatrix}\right) \\
&= \begin{pmatrix}0.53 & 0.06 & 0.41\end{pmatrix} \\
\mathbf{y}_{4,\text{decoded}} &:= \text{vocab}[\text{argmax}(\mathbf{y}_4)] = \texttt{<eos>}
\end{aligned}
$$

The decoded sequence is **`b, b, a, <eos>`**.