Introduction to Deep Learning for Speech and Text Processing
Exercise Sheet 10: Tricks & Sequence-to-Sequence Models

Thang Vu

8th January 2026

Dropout

When we apply dropout on the outputs of a layer during training, we need to scale the weights at test time. This is because we need the expected value of the layer outputs in the test phase to be consistent with the expected value in the training phase, otherwise the learned parameters of the network will be useless. The expected value of a random variable is similar to the mean but in a stochastic setting. For a random variable x with $\{x_1...x_m\}$ outcomes, each with probability $\{p(x_1)...p(x_m)\}$, the expectation $E(x)$ is defined as $\sum_{i=1}^{m} p(x_i)x_i$. This corresponds to the probability-weighted average of all possible values of the random variable.

Exercise 1.

(1) Compute $E(x)$ of a neuron x with probability p to dropout during training.

(2) What scaling factor do you need at test time to obtain the same expectation of x?

(3) Can you think of a way to ensure the same expectation of x during training and testing without having to scale x at test time?

Seq2seq Architecture

Exercise 2.

You are given a Seq2seq model with the following specifications: Each input in the source sequence has 100 dimensions and each output in the target sequence has 150 dimensions. The recurrent functions in the encoder and decoder are vanilla RNNs with a hidden state of 300 dimensions. In the case of attention, the output of $\tanh(W_c[c_t, h_t] + b_c)$ has 50 dimensions. Assume W_a being a square matrix for sum attention.
Compute the number of parameters for each of the following Seq2seq variants:

(1) without attention

(2) with attention using dot product as a scoring function

(3) with attention using bilinear function as a scoring function

Machine Translation

Exercise 3.

In this exercise we will decode the output of a Seq2seq model for English → German machine translation. The Seq2seq model consists of an encoder and decoder, both with vanilla RNNs that have ReLU activations. The decoder uses a dot product attention mechanism.
The input to the encoder is the sequence 'a christmas tree ⟨EOS⟩', where ⟨EOS⟩ is a special item of the vocabulary denoting the end of the sequence. The encoder outputs are already computed and are as follows:
$\overline{h_1} = \begin{bmatrix}1\\1\end{bmatrix}, \overline{h_2} = \begin{bmatrix}1\\0.5\end{bmatrix}, \overline{h_3} = \begin{bmatrix}0.5\\0\end{bmatrix}, \overline{h_4} = \begin{bmatrix}0\\0\end{bmatrix}$
The decoder has the following weight matrices, all bias vectors are 0:

$$W_\mathrm{i} = \begin{bmatrix} 0 & -1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}, W_\mathrm{h} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}, W_\mathrm{c} = \begin{bmatrix} 0 & 1 & 1 & 2 \\ 1 & -1 & 0.5 & 1 \end{bmatrix}, W_\mathrm{o} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.9 & 0 \\ 0 & 1 \end{bmatrix}.$$

The output vocabulary is V $= \{v_0 : \langle\mathrm{SOS}\rangle,\ v_1 : \langle\mathrm{EOS}\rangle,\ v_2 : \mathrm{ein},\ v_3 : \mathrm{Weihnachtsbaum}\}$.

(1) Compute the first two decoder outputs. To start decoding, we give a special start of sentence token $\langle\mathrm{SOS}\rangle$ as input with $y_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$. Always round to two decimals.

(2) When does the decoding process end?