# Deep Learning Assignment 2

**Kevin de Vries (10579869)**
University of Amsterdam
kevin.devries@student.uva.nl

## Question 1

### Question 1.1

For this question I use the notations $U$,$V$ and $W$ for $W_{hx}$, $W_{ph}$, and $W_{hh}$ respectively.

Gradient of $\mathcal{L}^{(T)}$ w.r.t $\hat{y}^{(T)}$:

$$\frac{\partial \mathcal{L}^{(T)}}{\partial \hat{y}_i^{(T)}} = \frac{\partial}{\partial \hat{y}_i^{(T)}} \left( -\sum_j y_j^{(T)} \log(\hat{y}_j^{(T)}) \right) = -\sum_j y_j^{(T)} \frac{\delta_{ij}}{\hat{y}_i^{(T)}} = -\frac{y_i^{(T)}}{\hat{y}_i^{(T)}} \tag{1}$$

Gradient of $\hat{y}^{(T)}$ w.r.t $p^{(T)}$:

$$\begin{aligned}
\frac{\partial \hat{y}_i^{(T)}}{\partial p_j^{(T)}} &= \frac{\partial}{\partial p_j^{(T)}} \left( \text{softmax}(p_i^{(T)}) \right) = \frac{\partial}{\partial p_j^{(T)}} \left( \frac{\exp(p_i^{(T)})}{\sum_k \exp(p_k^{(T)})} \right) \\
&= \frac{\partial}{\partial p_j^{(T)}} \left( \exp(p_i^{(T)}) \right) \frac{1}{\sum_k \exp(p_k^{(T)})} + \frac{\partial}{\partial p_j^{(T)}} \left( \frac{1}{\sum_k \exp(p_k^{(T)})} \right) \exp(p_i^{(T)}) \\
&= \frac{\delta_{ij} \exp(p_i^{(T)})}{\sum_k \exp(p_k^{(T)})} - \frac{\exp(p_i^{(T)}) \exp(p_j^{(T)})}{\left( \sum_k \exp(p_k^{(T)}) \right)^2} = \text{softmax}(p_i^{(T)}) \left( \delta_{ij} - \text{softmax}(p_j^{(T)}) \right) \\
&= \hat{y}_i^{(T)} \left( \delta_{ij} - \hat{y}_j^{(T)} \right)
\end{aligned} \tag{2}$$

Gradient of $p^{(T)}$ w.r.t $h^{(T)}$:

$$\frac{\partial p_i^{(T)}}{\partial h_j^{(T)}} = \frac{\partial}{\partial h_j^{(T)}} \left( \sum_n V_{in} h_n^{(T)} \right) = V_{ij} \tag{3}$$

Gradient of $h^{(t)}$ w.r.t $h^{(t-1)}$:

Preprint. Work in progress.

$$\begin{aligned}\frac{\partial h_i^{(t)}}{\partial h_j^{(t-1)}} &= \sum_k \frac{\partial h_i^{(t)}}{\partial a_k^{(t)}} \frac{\partial a_k^{(t)}}{\partial h_j^{(t-1)}} = \sum_k \frac{\partial}{\partial a_k^{(t)}} \left(\tanh(a_i^{(t)})\right) \frac{\partial}{\partial h_j^{(t-1)}} \left(\sum_l W_{kl} h_l^{(t-1)}\right) \\ &= \sum_k \delta_{ik} \left(1 - (\tanh(a_i^{(t)}))^2\right) W_{kj} = \left(1 - (h_i^{(t)})^2\right) W_{ij}\end{aligned} \tag{4}$$

Gradient of $\mathcal{L}^{(T)}$ w.r.t $p^{(T)}$:

$$\frac{\partial \mathcal{L}^{(T)}}{\partial p_j^{(T)}} = \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial \hat{y}_i^{(T)}} \frac{\partial \hat{y}_i^{(T)}}{\partial p_j^{(T)}} = -\sum_i y_i^{(T)} \left(\delta_{ij} - \hat{y}_j^{(T)}\right) \tag{5}$$

Gradient of $\mathcal{L}^{(T)}$ w.r.t $V$:

$$\begin{aligned}\frac{\partial \mathcal{L}^{(T)}}{\partial V_{jk}} &= \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial p_i^{(T)}} \frac{\partial p_i^{(T)}}{\partial V_{jk}} = \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial p_i^{(T)}} \delta_{ij} h_k^{(T)} = \frac{\partial \mathcal{L}^{(T)}}{\partial p_j^{(T)}} h_k^{(T)} \\ \Rightarrow \frac{\partial \mathcal{L}^{(T)}}{\partial V} &= \frac{\partial \mathcal{L}^{(T)}}{\partial p^{(T)}} \left(h^{(T)}\right)^T\end{aligned} \tag{6}$$

Gradient of $\mathcal{L}^{(T)}$ w.r.t $h^{(T)}$:

$$\begin{aligned}\frac{\partial \mathcal{L}^{(T)}}{\partial h_j^{(T)}} &= \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial p_i^{(T)}} \frac{\partial p_i^{(T)}}{\partial h_j^{(T)}} = \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial p_i^{(T)}} V_{ij} \\ \Rightarrow \frac{\partial \mathcal{L}^{(T)}}{\partial h^{(T)}} &= V^T \frac{\partial \mathcal{L}^{(T)}}{\partial p^{(T)}}\end{aligned} \tag{7}$$

Gradient of $\mathcal{L}^{(T)}$ w.r.t $h^{(t<T)}$:

$$\begin{aligned}\frac{\partial \mathcal{L}^{(T)}}{\partial h_j^{(t<T)}} &= \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial h_i^{(t+1)}} \frac{\partial h_i^{(t+1)}}{\partial h_j^{(t)}} = \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial h_i^{(t+1)}} \left(1 - (h_i^{(t+1)})^2\right) W_{ij} \\ \Rightarrow \frac{\partial \mathcal{L}^{(T)}}{\partial h^{(t<T)}} &= W^T \text{diag}\left(1 - (h^{(t+1)})^2\right) \frac{\partial \mathcal{L}^{(T)}}{\partial h^{(t+1)}}\end{aligned} \tag{8}$$

Gradient of $\mathcal{L}^{(T)}$ w.r.t $W$:

$$\begin{aligned}\frac{\partial \mathcal{L}^{(T)}}{\partial W_{jk}} &= \sum_t \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial h_i^{(t)}} \frac{\partial h_i^{(t)}}{\partial W_{jk}^{(t)}} = \sum_t \sum_i \frac{\partial \mathcal{L}^{(T)}}{\partial h_i^{(t)}} \delta_{ij} h_k^{(t-1)} \left(1 - (h_i^{(t)})^2\right) \\ \Rightarrow \frac{\partial \mathcal{L}^{(T)}}{\partial W} &= \sum_t \text{diag}\left(1 - (h^{(t)})^2\right) \frac{\partial \mathcal{L}^{(T)}}{\partial h^{(t)}} \left(h^{(t-1)}\right)^T\end{aligned} \tag{9}$$

Where $W^{(t)}$ is a dummy variable which is a copy of $W$, but which is only used at time step $t$. This is done to make computing the gradient simpler, since both $h^{(t-1)}$ and $W$ are dependent on $W$.

From the expressions for $\frac{\partial \mathcal{L}^{(T)}}{\partial W} = \frac{\partial \mathcal{L}^{(T)}}{\partial W_{hh}}$ and $\frac{\partial \mathcal{L}^{(T)}}{\partial V} = \frac{\partial \mathcal{L}^{(T)}}{\partial W_{ph}}$ I observe that $\frac{\partial \mathcal{L}^{(T)}}{\partial W_{ph}}$ is only dependent on the hidden state in last time step while $\frac{\partial \mathcal{L}^{(T)}}{\partial W_{hh}}$ is dependent on the hidden states at every time step. Since the chain rule has to be used for the full unrolled recurrent network, the recurrent network can easily suffer from vanishing or exploding gradients during backpropagation.

**Question 1.2**

The vanilla RNN was implemented using the boilerplate code and requirements given for the assignment. Since the assignment was ambiguous about the initialization of the parameters, the weight matrices were initialized by sampling from a normal distribution with a mean of $\mu = 0$ and a standard deviation of $\sigma = 0.001$. The biases and initial hidden state were initialized as zero vectors. The optimal initial hidden state is not learned during training in this implementation.

**Question 1.3**

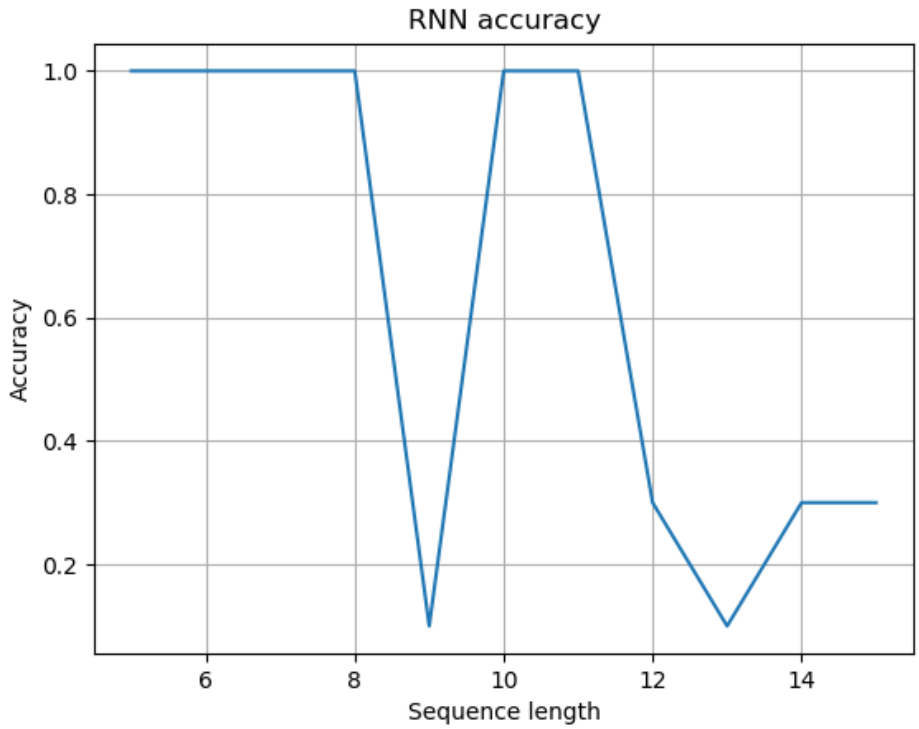


Figure 1: Accuracies reached by the RNN on palindrome lengths between sizes of 5 and 15

The RNN seems to be able to overfit very well to sequences of up to 10 characters with the exception of the sequence of 9 characters. From 11 characters onwards, the RNN seems to become unable to predict the last character, which shows that it is unable to memorize sequences which are too long. The values in Figure 1 are mostly estimated values if they are not equal to 1.00, since most of the sequence lengths did not converge to a single value, but were rather unpredictable.

**Question 1.4**

There are multiple benefits of using RMSprop and Adam. When using vanilla stochastic gradient descent (SGD), the gradients can start oscillating due to the curvature of the loss landscape when some parameters converge faster than others. Using momentum, the oscillations from these pathological curvatures in the loss landscape are dampened, because the optimizer maintains the momentum from previous parameters. Dampening the oscillations makes the gradients more robust, which in turn speeds up covergence. The optimizers also feature adaptive learning rates where aside from a static learning rate, previous gradients are used to make updates tamer (for large gradients) or more aggressive (for small gradients) depending on the local curvature of the loss surface.

**Question 1.5 a)**

The input modulation gate $g^{(t)}$ contains the information which is used as the hidden state in a vanilla RNN. The tanh activation function is thus used for the same reason why it is used in a vanilla RNN. This gate is multiplied by the input gate $i^{(t)}$ to choose which information should be stored in the cell state $c^{(t)}$. The forget gate $f^{(t)}$ also chooses which information to forget and remove from the cell state. Finally the output gate $o^{(t)}$ decides which information to read from the cell state into the hidden state $h^{(t)}$. Since $i^{(t)}$, $f^{(t)}$ and $o^{(t)}$ decide how much of the information to store, forget or read from the cell state, an activation between 0 and 1 is needed. This makes the sigmoid function $\sigma(\cdot)$ the most appropriate activation for these gates.

**Question 1.5 b)**

The total number of trainable parameters in the LSTM cell is given by:

$$4 \cdot (n \cdot d + n^2 + n)$$

This excludes the linear layer over the output of the LSTM cell $p^{(t)}$, since multiple LSTM cells can be stacked before applying $p^{(t)}$ over the output of the last cell. The amount of parameters in the LSTM cell is four times larger than in a vanilla RNN cell because of the input, forget and output gates applied to the cell state.

**Question 1.6**

The LSTM was implemented using the boilerplate code and requirements given for the assignment. The parameters are initialized in the same way as with the vanilla RNN. Additionally, the initial cell state $c^{(0)}$ is initialized with a zero vector and is not learned during training.

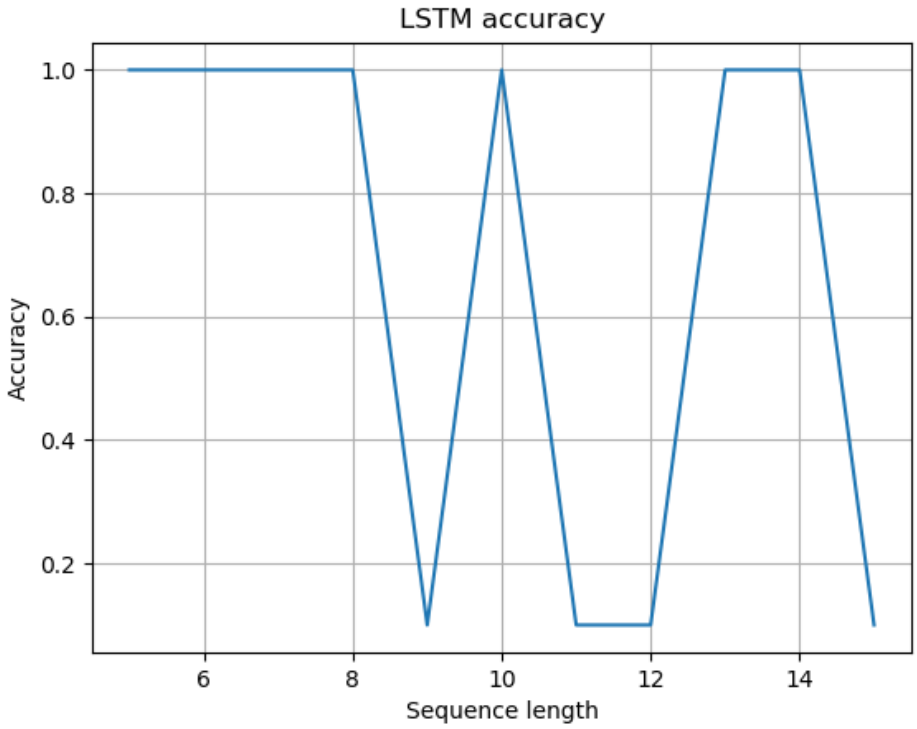


Figure 2: Accuracies reached by the LSTM on palindrome lengths between sizes of 5 and 15

The LSTM also seems to be able to overfit very well to sequences of up to 10 characters. From 11 characters onwards, my implementation also seems to suffer from the same issues as the RNN except

for the sequence lengths of 13 and 14 characters. This is probably due to the learning rate, which I did not manage to adjust on time to plot the correct curves. Again, lower values are mostly estimations, since these values did not converge to any particular values.

# Question 2

## Question 2.1

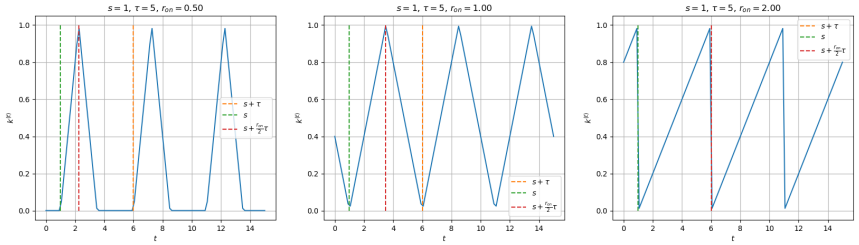


Figure 3: Visualizations of the temporal gate $k^{(t)}$ with $s = 1$, $\tau = 5$ and $r_{on}$ equal to 0.5, 1.0 and 2.0 respectively.

For question 2.1 the conceptual drawing has been replaced by three plots given in Figure 3 which display how the parameters $s$, $\tau$ and $r_{on}$ influence the behaviour of the temporal gate. The plots use a continuous $t$ instead of a discrete $t$ to give a more intuitive picture of the behaviour of the gate and its parameters.

## Question 2.2

Since the temporal gate $k^{(t)}$ only lets information through within some time frame for every $\tau$ steps, the LSTM can focus on the local temporal dynamics within a sequence using this gate. The temporal gate can thus be used for training on datasets where the temporal dynamics are highly localized.

## Question 2.3

The $s$ parameter shifts the periodic behaviour of the temporal gate and the $\tau$ parameter defines the time for a single period to complete. These parameters cannot be learned from the data since these parameters are discrete and thus not differentiable. The $r_{on}$ parameter influences when in the cycle and how much the information in the hidden and cell states is updated. This parameter can have a real value and since $k^{(t)}$ is at least piece wise differentiable with respect to $r_{on}$, $r_{on}$ can be learned from the data.

# Question 3

For every generated sentence line breaks have been replaced by spaces to make is easier to show the sentence in a latex generated pdf.

## Question 3.1 a)

For this question a two-layered LSTM has been implemented which is trained on “The picture of Dorian Gray” by Oscar Wilde. The model was trained for two epochs on sequences of length $T = 30$ where one epoch is equal to sampling as many characters as the size of the dataset. The model first transforms the data using an embedding layer to try and remove the correlations in the data caused by the indices of the characters that are used as input. The model was also trained using a dropout layer on the output of the second LSTM cell (which has remained unused when training) and a learning rate scheduler, which reduces the learning rate with a factor 0.96 every 1000 backpropagation steps. The training was done starting with the default learning rate of 0.002 given in the boilerplate code for the assignment. The number of hidden units and batch size were also the default values. During

training the gradients were also clipped using the norm of the gradients in order to prevent exploding gradients. After two epochs, the accuracy converged to a value around 0.65.

**Question 3.1 b)**

When training the model, sentences with a length of 30 characters were sampled by randomly sampling a character from the character set that the book contains and using it to predict the next character. The predicted characters were then added to the sentence and used to predict the next character in the sequence until a length of 30 characters was reached. The next character was determined as the character with the highest probability in the output.

The training starts out by generating sentences with relatively random probabilities. After 20 iterations, the sentence

3t a a a a a a a a a a a a a a

Is generated, which indicates that the model has learned the most prevalent characters in the English language. At 120 iterations, the sentence

Yer the was the was the was th

is generated. This indicates that the model has learned some of the most prevalent words in the English language. At 220 iterations, the sentence

ü "I had the some the some th

is generated. Now the model is learning more words and seems to be learning the concept of dialogue in the text. At 320 iterations, the sentence

Basil of the was the said the

is generated. Now the model seems to have learned some of the characters' names in the book. At 420 iterations, the sentence

é. I am that the seemed the se

is generated. The model seems to also have gotten an idea about punctuation. Finally at 520 iterations, the sentence

F I am so must the seemed the

is generated. The model seems to get stuck in a loop quite often due to the greedy sampling. In general, as the training progresses, the model starts learning more words and learns to create better combinations of words, but still keeps getting stuck in loops of subsequent words even at the end of the training.

**Question 3.1 c)**

After implementing the random sampling with temperature, a few sentences are sampled using temperatures of 0.5, 1.0 and 2.0 in order to compare them.

Using a temperature of 2.0 the sentences

y youxan! You sucles you?") Th

ProH-fouthe; Sputh.'"op Cloor.

Bellac_ of horrards self-shibl

are generated. These sentences seem to contain many spelling mistakes which make the sentences meaningless. Using a temperature of 1.0 the sentences

ything to do with anyone recal

Â who collector, the gains tha

zes one chapmer to you riding

are generated. These sentences seem to be much more meaningful, although they contain some spelling mistakes. Using a temperature of 1.0 the sentences

r the terror and passed with t

man's monstrous against the gr

! the thing the shallow passed

are generated. These sentences seem to be the best as they include enough variation to prevent that the model gets stuck in a word loop, but do not contain as many spelling mistakes.

**Bonus Question 3.2**

By sampling longer sequences as specified in the assignment and using the model to complete the sentence "Dorian saw the picture" the model generated the following sentence:

Dorian saw the picture in heavens! I wonder what it was the mention to the box all the past time before him. The screen the month dull common the artist had it intensely what to say the lad, stained on the painter's most mere dark to do with a strange of the bright chapters with the secret of same grotesque sins of the world calls of the painter fellow, more extremely charming monstrous was the only the senses of the top of the painter's mouth. There is nothing to be a rest of the room, and the