Recurrent Neural Networks

Yiangos Georgiou
11807288
University of Amsterdam
ygeorgiou12@gmail.com

1 Vanilla RNN versus LSTM

1.1 Vanilla RNN in PyTorch

1.1.1 Question 1.1

RNN and loss equations:

$$
\begin{aligned}
h^{(t)} &= tanh(W_{hx}x^{(t)} + W_{hh}h^{(t-1)} + b_h) \\
p^{(t)} &= W_{ph}h^{(t)} + b_p \\
\hat{y}^{(t)} &= softmax(p^{(t)}) \\
Loss &= -\sum_{k=1}^{K} y_k * log(\hat{y}_k)
\end{aligned}
$$

In order to compute the $\frac{\partial L^(T)}{\partial W_{ph}}$ we need to apply the chain rule and backpropagate the error of the loss.

1) Softmax Derivative
Since soft-max is a $^N \rightarrow^N$ function, the most general derivative we compute for it is the Jacobian

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

matrix.

$$\frac{\partial x_i^{(N)}}{\partial \hat{x}_j^{(N)}} \Rightarrow \frac{\partial(\frac{\exp(\hat{x}^{(N)}}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})})}{\partial \hat{x}^{(N)}}$$

$$\Rightarrow \frac{(\exp(\hat{x}_i^{(N)}))\prime * \sum_{k=1}^{dN}\exp(\hat{x}^{(N)}) - \exp(\hat{x}_i^{(N)}) * (\sum_{k=1}^{dN}\exp(\hat{x}^{(N)}))\prime}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})^2}$$

$$\Rightarrow \frac{(\exp(\hat{x_i}^{(N)}))\prime * \sum \hat{x}^{(N)} - \exp(x_i) * \exp(x_j)}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})^2}$$

$$\text{if i==j} \Rightarrow \frac{\exp(\hat{x}_i^{(N)}}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})} \frac{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)}) - \exp\hat{x}_j}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})}$$

$$\Rightarrow Softmax(\hat{x}_i)(1 - Softmax(\hat{x}_j))$$

$$\text{if i!=j} \Rightarrow \frac{(\exp(\hat{x_i}^{(N)}))\prime * \sum \hat{x}^{(N)} - \exp(x_i) * \exp(x_j)}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})^2}$$

$$\Rightarrow \frac{0 * \exp(x_j)}{\sum_{k=1}^{dN}\exp(\hat{x}^{(N)})^2}$$

$$\Rightarrow -Softmax(\hat{x}_i^{(N)}) * Softmax(\hat{x}_j^{(N)})$$

The jacobian matrix's dimensions are [N x N] where N is the number of outputs.

2) Derivatives with respect the W_{ph}

$$\frac{\partial L^{(T)}}{\partial W_{ph}} = \frac{\partial L^{(T)}}{\partial y^{(t)}} \frac{\partial y^{(t)}}{\partial p^{(t)}} \frac{\partial p^{(t)}}{\partial W_{ph}}$$

$$\frac{\partial L^(T)}{\partial y^{(t)}} = -\sum_{k=1}^{K} y_k * \frac{1}{\hat{y}}$$

$$\frac{\partial L^{(T)}}{\partial p_i^t} = \frac{\partial L^{(T)}}{\partial \hat{y}^{(t)}} \frac{\partial \hat{y}^{(t)}}{\partial p^{(t)}} \Rightarrow -\sum_{k=1}^{K} y_k * \frac{1}{\hat{y_k}} \frac{\partial \hat{y}^{(t)}}{\partial p^{(t)}} \Rightarrow -y_i * \frac{1}{\hat{y_i}} * \hat{y_i}(1 - \hat{y_i}) - \sum_{k \neq i} y_k * \frac{1}{\hat{y_k}}(-\hat{y_i}\hat{y_k})$$

$$\Rightarrow -y_i + y_i\hat{y}_i + \sum_{k \neq 1} y_k * (\hat{y}_i) \Rightarrow -y_i + \hat{y}_i$$

$$\frac{\partial L^{(T)}}{\partial W_{ph}} = \frac{\partial L(T)}{\partial y^{(t)}} \frac{\partial y^{(t)}}{\partial p^{(t)}} \frac{\partial p^{(t)}}{\partial W_{ph}} \Rightarrow (-y + \hat{y}) * \frac{\partial W_{ph} h^{(t)} + b_p}{\partial W_{ph}} \Rightarrow \underset{hidden_d \times 1}{(h^{(t)})} \underset{out_d \times 1}{(-y + \hat{y})^T} \Rightarrow \underset{hidden_d \times out_d}{\frac{\partial L(T)}{\partial W_{ph}}}$$

3) Derivatives with respect the W_{hh}

$$\frac{\partial L^{(T)}}{\partial W_{hh}} = \frac{\partial L^{(T)}}{\partial y^{(t)}} \frac{\partial y^{(t)}}{\partial p^{(t)}} \frac{\partial p^{(t)}}{\partial h^{(t)}} \frac{\partial h^{(t)}}{\partial h^{(0)}} \frac{\partial h^{(0)}}{\partial W_{hh}}$$

($\frac{\partial h^{(t)}}{\partial h^{(0)}}$) h_t depends on h_{t-1} and that itself depends on h_{t-2}, thus we have to propagate back and cover the error of the sequence to the h_0. You can think of this recursive procedure as an MLP with as many hidden layers as the input sequence size.

$$\frac{\partial h^{(c)}}{\partial h^{(c-1)}} = \frac{\partial h^{(t)}}{\partial (W_{hx}x^{(c)} + W_{hh}h^{(c-1)} + b_h)} \frac{\partial (W_{hx}x^{(t)} + W_{hh}h^{(c-1)} + b_h)}{\partial h^{(c-1)}} \Rightarrow (1 - (h^{(t)})^2) \frac{\partial (W_{hh}h^{(t-1))}}{\partial h^{(t-1)}}$$

$$\Rightarrow \underset{hidden_d \times hidden_d}{(W_{hh})_c^T} \underset{hidden_d \times 1}{(1 - (h^{(c)})^2)} \Rightarrow \underset{hidden_d \times 1}{\frac{\partial h^{(c)}}{\partial h^{(c-1)}}}$$

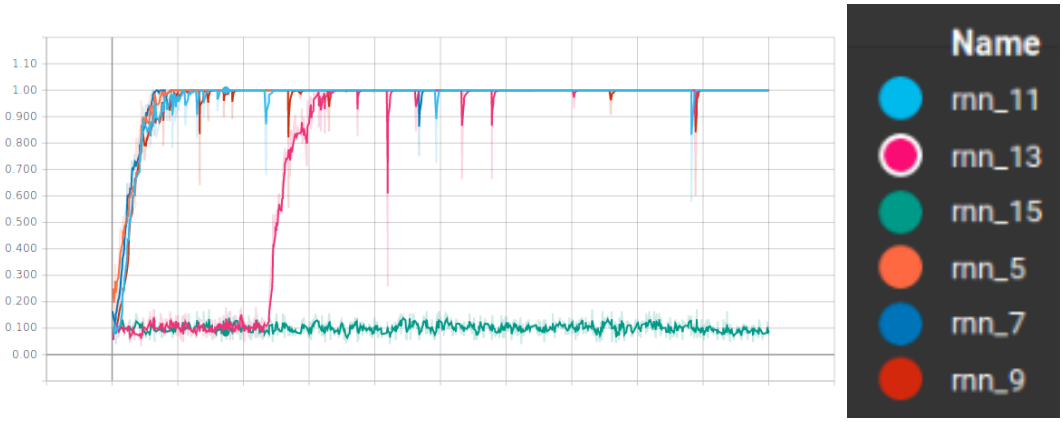

Figure 1: RNN accuracy for different sequence sizes(the name of each line indicates the size of the sequence).

Putting it all together

$$\frac{\partial L^{(T)}}{\partial W_{ph}} = \frac{\partial L^{(T)}}{\partial \hat{y}^{(t)}}\frac{\partial \hat{y}^{(t)}}{\partial p^{(t)}}\frac{\partial p^{(t)}}{\partial h^{(t)}}\frac{\partial h^{(t)}}{\partial h^{(0)}}\frac{\partial h^{(0)}}{\partial W_{hh}} \Rightarrow \underset{out_d \times 1}{(-y+\hat{y})}\frac{\partial p^{(t)}}{\partial h^{(t)}}\frac{\partial h^{(t)}}{\partial h^{(0)}}\frac{\partial h^{(0)}}{\partial W_{hh}}$$

$$\Rightarrow \underset{out_d \times hidden_d}{(W_{ph})^T} \cdot \underset{out_d \times 1}{(-y+\hat{y})}\frac{\partial h^{(t)}}{\partial h^{(0)}}\frac{\partial h^{(0)}}{\partial W_{hh}} \Rightarrow \underset{hidden_d \times 1}{\delta_l}\prod_{t=1}^{T}(\underset{hidden_d \times hidden_d}{(W_{hh})_c^T}\underset{hidden_d \times 1}{(1-(h^{(c)})^2)})\frac{\partial h^{(0)}}{\partial W_{hh}}$$

$$\Rightarrow \underset{hidden_d \times 1}{\delta_h} \cdot \underset{hidden_d \times 1}{h^{(0)}} \Rightarrow \underset{hidden_d \times 1}{h^{(0)}}\underset{hidden_d \times 1}{(\delta_h)^T} \Rightarrow \underset{hidden_d \times hidden_d}{\frac{\partial L^{(T)}}{\partial W_{ph}}}$$

The difference between the two gradients is that for the latter you have to apply the chain rule between hidden states for the whole sequence in order to back propagate the error of the loss. The main issues that might occur for long distance are vanishing and exploding gradients.

Vanishing gradients occur when the gradients have values less than one, thus, for very deep networks the product between all the consecutive steps from a specific sequence results nearly to 0. Due to that issue we have no derivatives to improve or change the model parameters.

Exploding gradients is exactly the opposite, when out gradients are bigger than 1, the product between the steps results into a huge gradient.

1.2 Question 1.3

For the experiments the default parameters lengths were used but with different sequence in order to see the capabilities of rnn memory.

As we can observe on figures [1] and [2], RNNs cannot learn palindrome sequence that exceed the size of 15 numbers. RNN memory has limitation on the number of symbols or numbers that can encode and remember.

1.2.1 Question 1.4

RMSProb tries to dampen the oscillations but in different way than momentum that is used in vanilla stochastic gradient descend, also, takes away the need to adjust learning rate while it does it automatically. Thus, this optimizer chooses a different learning rate for each parameter by calculating the exponential average of squares of gradients for each parameter, exponential because the weigthage of the previous term falls exponentially. The difference of RMSProb is that if we are heading towards the minimal the step size will automatically decrease. In that way optimal weights will stay the same while non-optimal weights will continue with higher learning steps.

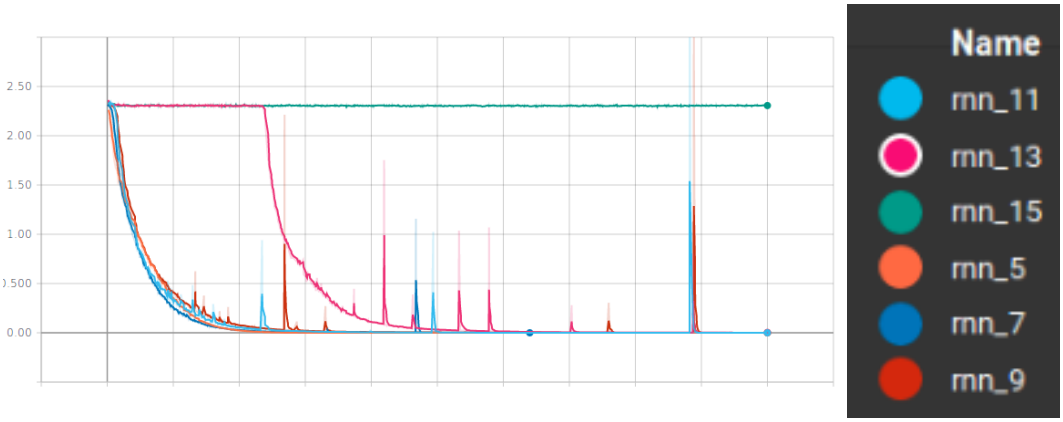

Figure 2: RNN loss for different sequence sizes(the name of each line indicates the size of the sequence).

Adam is another method that computes adaptive learning rates for each parameter. It an exponentially decaying average of past squared gradients RMSprop, however, Adam combines the heuristics of both Momentum and RMSProp. The exponential average of the gradient is also computed, thus to adapt the learning rate we multiply the learning rate with the average of gradients as was for the momentum and we divide it by the square average of gradients as in RMSProb.

The main difference between SGD and those two variation is the adaptive learning rate. That fact give to the model the capability to adapt to specific information or task and achieve better results.

1.3 LSTM

1.3.1 Question 1.5.a

- Input modulation gate : Creates the new candidate values. The output range is [-1,1] which output is added with input gate's output in order to increase or decrease a corresponding value in the cell state.
- Input gate : Is a fully connected MLP with sigmoid activation and its purpose is to decide which values is relevant with this time step. Sigmoid is used as a valve that allow the right amount of information from each value to pass through the node. The output range between [0,1], if the output value is small it indicates that the corresponding information should not pass through the network and vice versa.
- Forget gate : it looks at h_{t-1} and x_t and its purpose is to decide which information we have to forget. It works almost the same as the input gate but the difference is that the output of this gate is multiplied by the previous cell state(c_{t-1}) in order to forget what we don't need, while the input gate's output is multiplied by the new state in order to boost valuable information and reduce the weightiness of the useless information of the new input.
- Output gate : This gate purpose is to decide what parts of the cell state(c_t) we are going to output. It works exactly like forget and input gate(fully connected MLP, sigmoid activation) the difference is that its output is multiplied by the current cell state to remove any information that we don't want to output

1.3.2 Question 1.5.b

Each gate has input to hidden($d \times n$) and hidden to hidden($n \times n$) weights and n bias. Then a fully connected MLP follows with hidden to output($n \times d$ i assume that the same one hot vector which has d dimensions is used for both input and output) weights and d biases.

$$(4 * ((d * n) + (n * n) + n)) + (n * d) + d$$

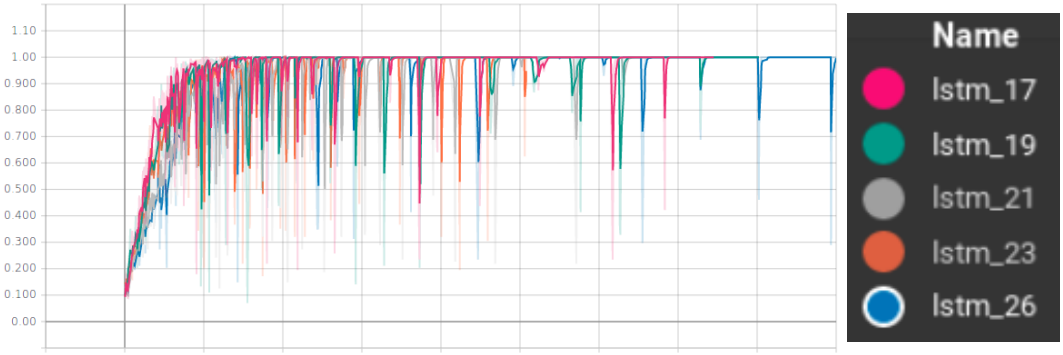

Figure 3: LSTM accuracy for different sequence sizes.(the name of each line indicates the size of the sequence)

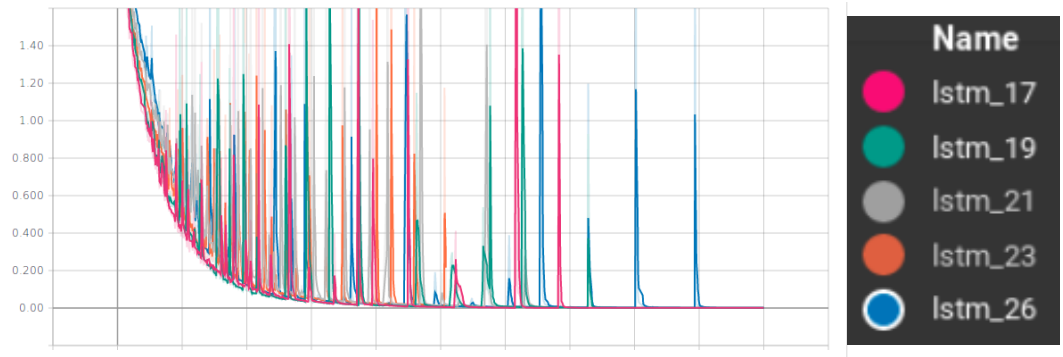

Figure 4: LSTM loss for different sequence sizes.(the name of each line indicates the size of the sequence)

1.3.3 Question 1.6

Regarding figures [3] and [4], we see that the capabilities of a LSTM is way bigger than RNN, its complicated architecture makes able to encode and remember information that occurred many steps before.

2 Modified LSTM Cell

2.1 Question 2.1

In order to examine how temporal gate $k^{(t)}$ changes over time we monitor how the values change in a toy example. This gate function by using 3 parameters, which are initialized as $\tau = 10, \sigma = 3$ and $r_{on} = 0.4001$.

2.2 Question 2.2

Considering the previous toy example(table 1) and the controlled gate updates that extend LSTM with an additional gate(k). Update equation:

$$c^{(t)} = k^{(t)} * \hat{c^{(t)}} + (1 - k^{(t)}) * c^{(c-1)}$$

Give that update equation and considering the results on table 1 we can see that this gate has two phases, one that it's closed where $k^{(t)} = 0$ and as a result to not change the cell state and one that it's open, where $k^{(t)} > 0$ and the cell state changes during this phase. Moreover, as we can see in figure 5 the open phase has two phases one that increasing the influence of the current state where

t	$\phi^{(t)}$	$\phi^{(t)} < \frac{r_{on}}{2}$	$r_{on} > \phi^{(t)} > \frac{r_{on}}{2}$	$k^t value$
1	0.8	False	False	0
2	0.9	False	False	0
3	0	True	False	0
4	0.1	True	False	0.5
5	0.2	True	False	1
6	0.3	False	True	0.5
7	0.4	False	True	0
8	0.5	False	False	0
9	0.6	False	False	0
10	0.7	False	False	0
11	0.8	False	False	0
12	0.9	False	False	0
13	0.0	True	False	0

Table 1: Toy example for gate $k^{(t)}$

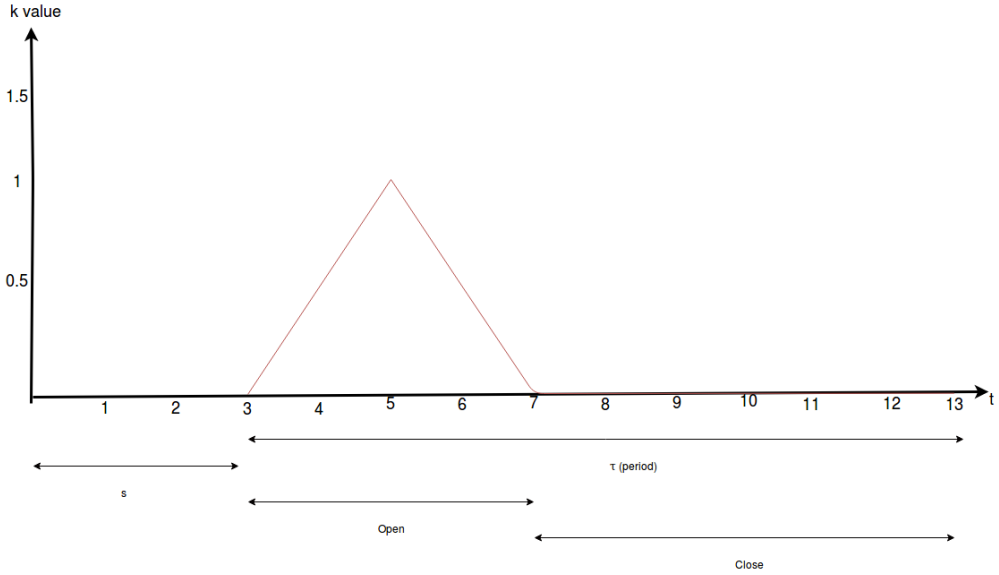

Figure 5: Value change over time for gate k

$k^{(t)}$ values go from 0 to 1(timesteps 3-5 in figure 5), then it follows the phase that decreasing the influence of the current state(timesteps 5-7 in figure 5).

What can be beneficial with this additional gates is that the units only update during the open phase time instead of every step, thus we ll get fewer update steps. Moreover,you can achieve a longer and adjustable memory length through τ.

In other words, this gate decides when the gradient is required for the procedure,in that way it allows an undecayed gradient to be backpropagated, thus our model converge faster.

2.3 Question 2.3

The parameters of the gate can be explained as follows:

- τ is the length of a cycle
- s is the delay that your gate is activated regarding the timesteps.
- r_{on} is the variable that controls the duration if the open phase.

The parameters that can be learned during training is τ, s and r_{on} because it's all differentiable.

3 Recurrent Nets as Generative Model

3.1 Question 2.1

To examine whether our model is able to generate its own sentences, we implement a procedure that pass into the network a random character and let the model to finish a sentence of size 30.

step	sentence	accuracy	loss
0	f	0%	4.515
400	Ved the king the king the king	45%	1.839
800	Ond the seast and said, ‘Wha	56%	1.444
1200	Marly the stream and the strea	58%	1.399
1600	You have a son with the stream	60%	1.300
2000	the second strength to be go	61%	1.264
2400	[and the second son went to t	62%	1.206
2800	’ said the cat, ‘I am not	63%	1.180
3200	And the cook said, ‘I will n	65%	1.132

Table 2: Generated sentence examples

As we can see, on the first steps the model can create words, yet those words cannot make any sense. That happens because model’s memory consider the previous 2 or 3 character to predict the next character. Moreover, the model tends to generate words that can be found many times in the given text. Regarding steps between 800-1600, we can observe that our model starts to remember and find correlation for more than 2-3 character as a result to get almost meaningfull sentences, yet those sentences shares words that we meet many times in the given text. On step 3200, we can observe syntax coherence between the words of the sentence.

Another aspect of to modify a model is to add a term that is called temperature in order to make it more or less greedy. We trained 2 identical networks, one with temperature 0.5 and one with 2. The difference that we observed is that for temperature 0.5 the training was faster and the generated sentences had similar content or maybe the same words. Regarding the model with temperature 2, the training is slow and the generated phrases contained more diverse vocabulary, however we get also sequence of letters that don’t assemble a word.

3.2 Bonus

For this part a random input(phrase of 30 character) were fitted into the network. Then the network purpose is to generate a paragraph of 1500 character.

- **Generated Text :** last of one of the giants. For a little tailor to the bottom of the garden, and was out of her bottom there lay a will was to be his Holiness the tree to watch; but about twelve o’clock had struck, the little bit and put them on his shoulders and remained sitting there, and he called out, ‘Don’t bring me.’ ‘I will come and asked for my brothers, the old woman seized him and scratched at all, and all four-footed animals were sitting over the fire, and the soldier said to the little man, ‘you are good for nothing,’ he answered, and held his leathern cap and said: ‘I will go to the bad of the well to seek the bottom of the garden, and when the king made the first twig, dear father will only lie in bed and found his thirst grew every day he went into the garden, and when the king made the first twig, dear father will only lie in bed and placed the cows with them. The man had not taken their feet for the third time, a court order to try it on the giants. The little tailor at last it looked backwards and forwards. And when she saw the second drank and drank to his heart to the ball. But her mother said to the little man. ‘Full threepence,’ cried the miller. ‘Why not?’ he replied, ‘I will go to my father and sound and sang–

  ‘My mother killed her little son; My father grieved when I was gone; My sister loved me best of all; She laid her kerchief over me, And took my bones that they might lie Underneath the juniper-tree Kywitt, Kywitt, what a beautiful bird am I!’ **: end**’
- **Generated Text :** id so, and when they were once sitting thus when they were out hunting, she was so well that he could not help setting out upon her plate, he took it into the air. The

bird pleased the soldier, and as she had subbose till she came to the top of the house there was a great feast. When the forester ran in a very scanty meal of missing of any money paid by a user who had been saved by any opposition to the town, and said to him, ‘What a pity one more piece of bread, and made a high to wipe so long that at last he let his booty go. As soon as they drew near the morrow! Who is bride until the great treasures, which were gone, and when they were once sitting on a throne. So he asked his kinsmen, and took her in his arms and companied at him with their fighting, and said: ‘Is it I, or is!’ Then he went out and asked the king to her father said they, ‘why should you wish for?’ He answered: ‘Do be a strange peasant called all his might. Then came a pair of public domain works in compliance with the first reason use and gathered them their father’s house. They gave him a common flowers.

This days as he was told about a long time afterwards, happy and draw what he was to shudder, but you will not do it, I will be king.’ So he said, ‘I will go and see the bearer, and was very angry, and said: ‘I will go down first, and looked at her face and hands, and the spit was sobed her time, he caused a while, and at last settled near a traveller, and hung her up in his hand, and saw the flower, and the s **: end**