Deep Learning Assignment 2

Nils Lehmann
12868175
nils.lehmann@student.uva.nl

1 Recurrent Neural Networks

1.1 Vanilla RNNs

1.1.a

$$\begin{aligned}\frac{\partial \mathcal{L}^{(T)}}{\partial \boldsymbol{W}_{ph}} &= \frac{\partial \mathcal{L}^{(T)}}{\partial \hat{y}^{(T)}} \frac{\partial \hat{y}^{(T)}}{\partial p^{(T)}} \frac{\partial p^{(T)}}{\partial \boldsymbol{W}_{ph}} \\ &= (y^{(T)} \frac{1}{\hat{y}^{(T)}}) \left(\frac{\partial}{\partial p^{(T)}} \frac{\exp p^{(T)}}{\sum_k \exp p_k} \right) h^{(T)}\end{aligned}$$

1.1.b

$$\begin{aligned}\frac{\partial \mathcal{L}^{(T)}}{\partial \boldsymbol{W}_{hh}} &= \sum_{i=0}^{T} \frac{\partial \mathcal{L}^{(T)}}{\partial \hat{y}^{(T)}} \frac{\partial \hat{y}^{(T)}}{\partial p^{(T)}} \frac{\partial^{(T)}}{\partial p_i} \frac{\partial p_i}{\partial \boldsymbol{W}_{ph}} \\ &= \sum_{i=0}^{T} \frac{\partial \mathcal{L}^{(T)}}{\partial \hat{y}^{(T)}} \frac{\partial \hat{y}^{(T)}}{\partial p^{(T)}} \left(\prod_{j=i+1}^{T} \frac{\partial p^j}{\partial p^{j-1}} \right) \frac{\partial}{\partial \boldsymbol{W}_{hh}} \boldsymbol{W}_{ph} \left(\tanh \left(\boldsymbol{W}_{\boldsymbol{hx}} x^i + \boldsymbol{W}_{hh} h^{i-1} + b_h \right) \right)\end{aligned}$$

1.1.c

The first gradient $\frac{\partial \mathcal{L}^{(T)}}{\partial \boldsymbol{W}_{ph}}$ does not depend on previous time steps, while the gradient for $\frac{\partial \mathcal{L}^{(T)}}{\partial \boldsymbol{W}_{hh}}$ contains a product over time steps T which implies that if T is large and the partial derivatives will evaluate to small values, the product of those terms will lead to a vanishing gradient problem.

1.2 Long Short-Term Memory (LSTM) network

1.2.a LSTM Gates

- Modulation gate $g^{(t)}$: This gate calculates the potential new values that would update the new cell state. The tanh function is a good choice because it normalizes values to be between -1 and 1, which is a desirable quality for better learning.
- Input gate $i^{(t)}$: This gate considers what values should be considered for an updated new cell state. By applying the sigmoid function, this gate essentially decides which values should be updated and to what degree between 0 (no update) to 1 (complete update). Thus, the input gates decides which cell state values to update, while the modulation provides values between -1 and 1 that would be used to update the new cell state.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

- Forget gate $f^{(t)}$: The forget gate decides what information from the current state is relevant to keep and what to dismiss. The sigmoid non-linearity is able to decide what information to throw away or keep because it outputs values between 0 and 1 and it can still be differentiated.
- Output gate $o^{(t)}$: Lastly, the output gate decides which elements should be considered as the new output at the current time step. Again a sigmoid function is a good choice to decide in what range between "discard all" or "keep all", this information should be considered.

1.2.b LSTM Trainable Parameters

$$4 * (N_{input} \times N_{hidden} + Nhidden \times N_{hidden} + N_{hidden}) + (N_{hidden} \times N_{output}) + N_{output}$$

1.3 LSTMs in Pytorch

The training loss and accuracy averaged over three random seeds for sequence lengths 5, 10, and 15 on the random combinations dataset are shown in Figure 1. The final mean and standard deviation over the random seeds on a separate test dataset of size 5000, is reported in Table 1. We can observe that as the sequences get longer, the LSTM struggles more to achieve perfect accuracy. Although it manages to do so very fast for sequence length five, it takes a little bit under 2000 iterations for a sequence length of 10. We can also observe that during training there is more variance around loss and accuracy measures when the sequence length is longer. On the testset, sequence length 10 has the largest standard deviation in results, while sequence length 15 has the largest standard deviation for accuracy. Both the LSTM and subsequent biLSTM experiment were conducted with the parameters given by the assignment. The embedding size was set to one.

Sequence Length	Average Loss	Std Loss	Average Accuracy	Std Accuracy
5	0.0011	0.0006	1.00	0.0
10	0.1625	0.0380	0.9983	0.0012
15	0.6945	0.0089	0.8273	0.0069

Table 1: LSTM results for different sequence lengths.

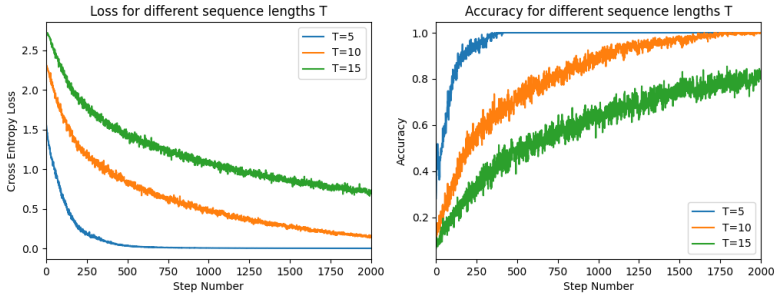

Figure 1: LSTM training loss and accuracy over random seeds for different sequence lengths T

1.4 Bidirectional LSTM

The training loss and accuracy curves of the Bidirectional LSTM, when compared to the vanilla LSTM, clearly demonstrate much faster convergence for all three sequence lengths. Unlike the vanilla LSTM, perfect accuracy is achieved in all cases after around 200, 750, and 1000 steps for the three sequence lengths 5,10, and 15 respectively. Although there remain some slight differences in the mean and standard deviation of the loss on the testset, the accuracy is perfect for all three cases with standard deviation of zero across the random seeds.

Sequence Length	Average Loss	Std Loss	Average Accuracy	Std Accuracy
5	0.0002	0.0001	1.00	0.0
10	0.0043	0.0008	1.00	0.0
15	0.0144	0.0007	1.00	0.0

Table 2: BiLSTM results for different sequence lengths.

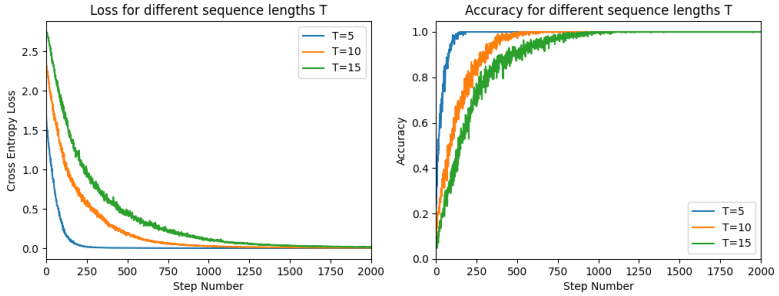

Figure 2: BiLSTM training loss and accuracy over random seeds for different sequence lengths T

2 Recurrent Nets as Generative Model

2.1 Two-Layer LSTM

2.1.a

A learning rate of 0.001 was chosen in combination with a cosine annealing learning rate scheduler that Phillip demonstrated in the Tutorials. In combination with the Adam optimizer this seemed an appropriate starting choice and produced acceptable results. The hidden dimension size was increased to 256 to increase model complexity and as suggested by the TAs, the embedding size was set to one half of the hidden size so 128. The max norm for gradient clipping was set to the default value 5, because it gave reasonable result and I was not aware of additional heuristics that would indicate the ideal value. The batch size was set to 128 as training could be done on the Lisa cluster and models were trained for initially 25000 steps. The main experiment was conducted with the book "The Brothers Karamazov" by Dostoevsky, but subsequent experiments also include "The Stranger" by Camus, in order to test the model with text corpora of varying size since text prediction outcomes can greatly depend on corpus size. Inspecting Figure 3, one can see that although the loss and accuracy improve, the main gains come in early step iterations, and there remains a fairly large degree in accuracy variance throughout subsequent training.

	lr	optimizer	hidden size	embed size	batch size	train steps	max norm
Karamazov	0.001	Adam	256	128	128	25000	5.0
Stranger	0.001	Adam	256	128	128	25000	5.0

Table 3: Hyperparameter choices for Two-Layer LSTM.

2.1.b Text Generation Greedy Sampling

In order to inspect randomly generated text samples from a variety of stages during training, table 4 contains samples from a model run on the full 25000 iterations, and samples where in a second training run, samples were gathered based on $\frac{1}{3}$, $\frac{2}{3}$, and $\frac{3}{3}$ of the final accuracy. The to be generated sequence length was set to 50 in order to also inspect coherency with respect to sequence length without rerunning experiments for different sample lengths (vertical bars were inserted to see the 30 character mark as postprocessing step). Looking at the development patterns in the right column, one can observe that at $\frac{1}{3}$ of final accuracy, the model mainly generates combinations of the vowels "a" and "e" and no readable words. Considering examples, when the model achieves $\frac{2}{3}$ or around 44%

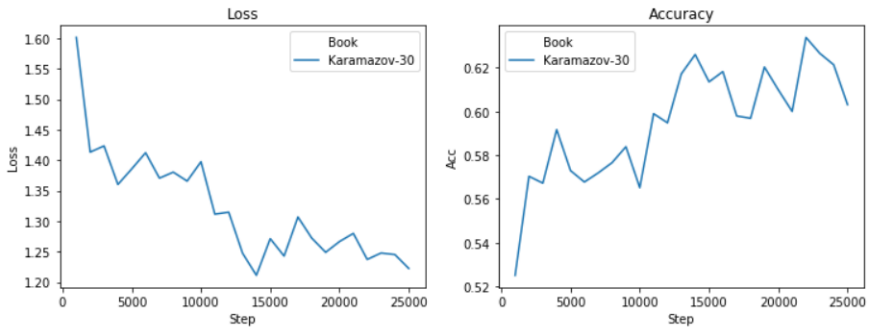

Figure 3: Training Loss and accuracy for "The Brothers Karamazov" with sequence length of 30

	Karamazov 25k training steps		Karamazov based on final accuracy of 0.65
$\frac{1}{3}$	Katya was a straight that he w**I**as a strained the sa n the same to the time to the **I**time to the time to Mitya was the same to the time**I** to the time to the 6 I was a strange to the time **I**to the time to the t ! I was the same to the time t**I**o the time to the ti	$\frac{1}{3}$	Xr a ae te ae te ae te ae te a**I**e te ae te ae te ae 8n a ae te ae te ae te ae te a**I**e te ae te ae te ae Wn a ae te ae te ae te ae te a**I**e te ae te ae te ae w a ae te ae te ae te ae te ae**I** te ae te ae te ae t àme te ae te ae te ae te ae te**I** ae te ae te ae te a
$\frac{2}{3}$	He was a strange that he was a**I** strange that he was g the money and the same thing**I**s are not a strange ou are a strange that the mone**I**y was a strange that ing the prosecutor was a stran**I**ge that he was a str VII. The Brothers I was a stra**I**nge that he was a st	$\frac{2}{3}$	and the was and the was and th**I**e was and the was an g the was and the was and the **I**was and the was and Kand the was and the was and t**I**he was and the was a ve the was and the was and the**I** was and the was and he the was and the was and the**I** was and the was and
$\frac{3}{3}$	0 a sort of the money and the **I**same time the same t $. I was a strange to say that**I** I was a strange to But the prosecutor was a stran**I**ge to say that he wa çe to the same time to the sam**I**e time to the same t 1. If an instant that he and was**I** a strange to say th	$\frac{3}{3}$	questions and the same time to**I** the same time to th VII. The Second the money and **I**the same time to the Quite a secret of the town wit**I**h the same time to t n the same time to the same t**I**Ime to the same time Æson, and the prosecutor was s**I**till and see him and

Table 4: Text generation samples at different training step iterations. Vertical bars demonstrate the sequence length at which the model was trained.

accuracy, the samples consist of coherent words, although it is mainly a repetition of "was", "the", and "and". When the model achieves its highest accuracy, we can observe samples that are more close to sentences, albeit we again see a repetition of the phrase "the same time". With respect to samples that are longer than the trained sequence length of 30, the samples seem to produce the same content across different samples, even though the sequence before the 30 character mark differs. Thus, in later stages of training, sequences of length shorter than 30 give different results across samples, but after the length 30 mark, the output is often the same. Another interesting observation is that given a non-alphabetical starting character like "6" or "!" in the left $\frac{1}{3}$ column, which effectively means the opportunity to begin a new sentence, the model generates almost the exact same sequence.

2.1.c Text Generation with Temperature

Since temperature is defined as the reciprocal of the temperature parameter τ, a low τ yields a high temperature, and a high τ yields a low temperature. With respect to the distribution from which we sample our next character, a high temperature will lead to a uniform distribution, while a low temperature will lead to a sharply peaked distribution which will have the same effect as applying the argmax. The impact of the τ parameter can clearly be observed in the samples generated in Table 5, where a τ value of 0.5 leads to random gibberish, line breaks, wrong capitalization and would probably make a good password generator. With τ equal to one, the sentences still do not make a lot of sense, however, the individual words mostly do. A τ value of 2, leads to similar results as table 4 samples with greedy sampling, where no misplaced line breaks occur, and words are put together in such a way that resemble readable sentences.

Temperature τ	Generated Text
0.5	Qutisten. Though Eluts it's just varilyemI,' he te
	fist, Mitya's most Smerdy walk—his!"—he_)..5(*4've
	, parple, what, Karta!' palt chose wephes; inselke
	do thEm pressed Him, 'What?, thosaboB Now not ...
	4/" Brine, Dmitri Kitrifox? Nalaxating! This was n
1	pose." She blutted him." Mitya was raised again.
	You see, too! Lost Smerdyakov bore!" he had at the
	$05n), there to call you all character, that's a w
	ît this present will turn the room on the door. E
	—an inquired that he was a passage. But us means t
2	) the word by the defense since he had seen him to
	Just had told me a man sent like a great deal for
	àw and say, at the time, and when he had come back
	ke the time for the time, that she had been taken
	our present excitement for the room. It was not a

Table 5: Text Generation samples from randomly initialized first character for different values of τ

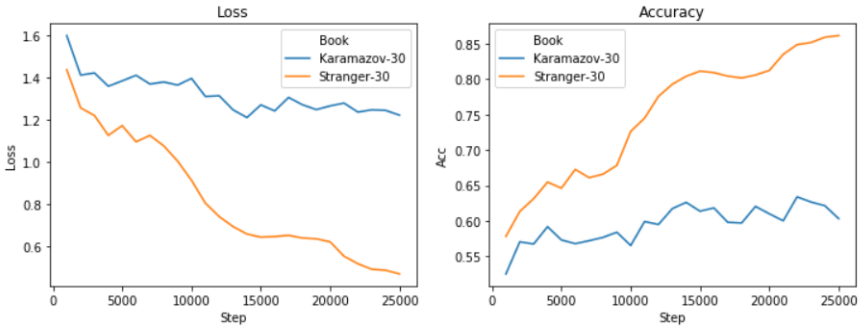

Figure 4: Training Loss and accuracy for "The Brothers Karamazov" and "The Stranger" trained on sequence length 30

2.2 Bonus

"The Stranger" by Camus is a much shorter book (only 197,206 characters, compared to 1,957,596 in Karamazov), and the LSTM achieves much higher accuracy here as it is likely significantly overfitting on the given data. The finished sentences, for Camus in table 6 show this in several ways: The phrase "the point was of quite minor importance" after a semicolon is "copied" exactly from the book. Similarly the phrase "blurred dark form wobbling in the heat haze" and "watery blue eyes" are written the book, however, in both examples the model continues in a different way. With respect to "The Brothers Karamazov", it is interesting to note that the model both times after encountering a comma in the input, chooses the phrase "if only I think of" as a next segment but then continues in different ways.

	Finished Sentences
Dostoevsky	I can see the sun, if only I think of the contrary, the prosecutor had a load and the same five or a shake of the peasants and the end of his property I think the devil, if only I think of the rest of the prisoner was about that very days are all things are not ready to do so, and he spoke and was gr He had stayed at the town. He was starting at him, and at last time to make it all after his father. And then she was speaking that he had been convin
Camus	I saw him as a blurred dark form wobbling in the heat haze, there was nothing very impressing about it, and it was moving link back and spaghetti, di He gave me a long look with his watery blue eyes. Then we shook hands wrong in insisting on this; the point was of quite minor importance, and, unle He had a wonderful coat; in fact, that was his best wishes; there was a good three quarters of an hour, I should say. Then a bell rang. My lawyer

Table 6: Finished sentences. Underlined segments are the inputs to the respective model. $\tau = 2$ was applied during sampling.

3 Graph Neural Networks

3.1 GCN Forward Layer

3.1.a

The propagation rule for a GCN layer comprises the term $\widetilde{A}$, which holds information about all connections for a particular node including the connection to itself. The features H^l of all nodes are multiplied by a weight matrix W^l which yields a message for each node. Subsequently, these messages are passed to neighboring nodes via the adjacency matrix $\widetilde{A}$ and averaged to generate an updated message per node.

3.1.b

Implicitly, the GCN layer assumes that all messages have equal importance, which means that nodes with same different inherit features but same neighbors can obtain exactly the same features. This is problematic because these nodes have become indistinguishable given their new features. A simple solution to overcome this issue is to introduce a weight matrix that weighs messages which a node passes to itself during message passing higher and thus preserves the unique node message to a greater extent. Or use attention.

3.2

3.2.a

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

3.2.b

Messages can be passed from C to E, either via D,B,A,E or via D,F,A,E. Either way, it will take 4 updates until information from node C arrives at node E.

3.3

The equation is missing the attention weights a_{ij}, which calculate all the attention values for a node i that is connected to a node j. Without this term, the update equation would not include the attention mechanism at heart of Graph Attention Networks.

$$h_i^{l+1} = \sigma\left(\sum_{j \in \mathcal{N}(i)} a_{ij} W^{(l)} h_j^{(l)}\right)$$

3.4

Since researchers find more and more ways to represent data in a way that a GNN can be effectively applied, the number of real world applications has significantly increased in recent years both in quantity and variety over. Zitnik et al. use a GCN to model interactions and side effects at molecule level. This model is designed to test whether certain drugs that could be mixed for better health outcomes are safe to use and are more cost effective than other tests that could find such relationships (Zitnik et al. [2018]). In an area that does not immediately stand out to be suited to GNNs, Wu et al. apply a GNN to multivariate time series forecasting and show that a GNN comes to par or outperforms statistical methods and other RNN architectures (Wu et al. [2020]).

3.5

3.5.a

Given a dataset in a sequence and graph representation, the performance of RNNs and GNNs respectively, would significantly depend on the inherit structure of the data and the task at hand. If the task were to identify causal relationships with many complex connections, the GNN through modelling the inherit structure of connections directly, would perform better than a RNN. Similarly,a GNN would also be able to better handle data that contains a lot of spatial information, such as images, because again it is able to exploit such information directly, whereas a RNN would have a more difficult time to learn from spatial data. I find it difficult to think of examples, where a RNN would clearly outperform a GNN, mainly because most data contains relational dependencies that a RNN is not able to learn because the time interval of accessing necessary information is too long for example. As previously mentioned, GNNs were even successfully applied in a mainly temporal setting with time series forecasting (Wu et al. [2020]). However, it might be more difficult to align data in such a way that a GNN can effectively use its advantages.

3.5.b

Zhao et al. develop a so called "Temporal Graph Convolutional Network", that combines a GCN and a GRU in order to make traffic pattern predictions (Zhao et al. [2019]). While, the GCN is used to learn spatial relationships for a road network, the output of that layer is fed to the GRU to make inferences about the changing traffic data over time. The combined output can subsequently be utilized to make predictions and suggestions for road navigation.

References

Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. *arXiv preprint arXiv:2005.11650*, 2020.

L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

M. Zitnik, M. Agrawal, and J. Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.