
Assignment 2. Recurrent Neural Networks and Graph Neural Networks

Maurice Frank
11650656
maurice.frank@posteo.de
Code: [github](#)

1 Vanilla RNN versus LSTM

1.1 RNN derivatives

Generally we have:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{ph}} = \frac{\partial \mathcal{L}}{\partial \mathbf{p}} \frac{\partial \mathbf{p}}{\partial \mathbf{W}_{ph}} \quad (1)$$

With the two partials:

$$\frac{\partial \mathcal{L}}{\partial p_i} = - \sum_j y_j \frac{\partial \log \hat{y}_j}{\partial p_i} \quad (2)$$

$$= - \sum_j \frac{y_j}{\hat{y}_j} \frac{\partial \hat{y}_j}{\partial p_i} \quad (3)$$

$$= -y_i(1 - \hat{y}_i) - \sum_{j \neq i} \frac{y_j}{\hat{y}_j} \cdot (-\hat{y}_i \hat{y}_j) \quad (4)$$

$$= -y_i + y_i \hat{y}_i + \hat{y}_i \sum_{j \neq i} y_j \quad (5)$$

$$= \hat{y}_i \left(y_i + \sum_{j \neq i} y_j \right) - y_i \quad (6)$$

$$= \hat{y}_i - y_i \quad (7)$$

$$\iff \quad (8)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{p}} = \mathbf{p} - \mathbf{y} \quad (9)$$

$$(10)$$

Note here it holds $\sum_i y_i = 1$ because of the one-hot encoding.

The second derivative is more direct:

$$\frac{\partial \mathbf{p}}{\partial \mathbf{W}_{ph}} = \mathbf{h}^{(T)} \quad (11)$$

$$(12)$$

leads finally to:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{ph}} = (\mathbf{p} - \mathbf{y}) \cdot \mathbf{h}^{(T)} \quad (13)$$

The derivative with respect to the hidden weight we write down in its recursive form:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{hh}} = \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{p}} \frac{\partial \mathbf{p}}{\partial \mathbf{h}^{(T)}} \frac{\partial \mathbf{h}^{(T)}}{\partial \mathbf{W}_{hh}} \quad (14)$$

$$\frac{\partial \mathbf{h}^{(T)}}{\partial \mathbf{W}_{hh}} = (1 - \mathbf{h}^{(T)^2}) \cdot \frac{\partial}{\partial \mathbf{W}_{hh}} \mathbf{W}_{hh} \mathbf{h}^{(T-1)} \quad (15)$$

$$= (1 - \mathbf{h}^{(T)^2}) \cdot \left[\left(\frac{\partial}{\partial \mathbf{W}_{hh}} \mathbf{W}_{hh} \right) \mathbf{h}^{(T-1)} + \mathbf{W}_{hh} \left(\frac{\partial}{\partial \mathbf{W}_{hh}} \mathbf{h}^{(T-1)} \right) \right] \quad (16)$$

$$= (1 - \mathbf{h}^{(T)^2}) \cdot \left(\mathbf{h}^{(T-1)} + \mathbf{W}_{hh} \frac{\partial \mathbf{h}^{(T-1)}}{\partial \mathbf{W}_{hh}} \right) \quad (17)$$

$$(18)$$

Because we had to write down $\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{hh}}$ recursively which we do not do for $\frac{\partial \mathcal{L}}{\partial \mathbf{W}_{ph}}$ we directly see the different length of the temporal dependencies in the two computational graphs. For the hidden weights gradient we need to transverse the whole time-sequence while the other one only ever depends on the final hidden state. This gives problems in practical training probable. The partial gradients of the hidden weights might be small and they form a product. The gradient might be vanishing. In practice this might go so far to reach the numerical limits of floating point arithmetics.

1.2 Vanilla RNN code

Find the code inside `vanilla_rnn.py` and `train.py`.

1.3 Vanilla RNN experiment

See Figure 1 for a overview plot of the results and Section 1.6 for a discussion/comparison of the results.

1.4 Optimizer

SGD has problems. One of them is occurring oscillations in valleys of the loss space. SGD does not have any *memory* and thus just tries to approximate the currents face's gradient to follow down which might make the path jump around a minimum of the valley. One change to counter this problem is introducing **momentum**. Following the intuition of the physical term, the gradient with momentum gets only changed gradually not sudden in every optimizer step. This is implemented as a decaying average of gradient updates. The weights get updated as a weighted sum of the previous update and the new gradient. A second idea is to tweak the learning rate for each weight and not use a fixed η for all, yielding a **adaptive learning rate**. For those weights that change a lot (bounce around some valley) we want to reduce the update step to counteract the bouncing. This can be seen in the RMSProp [1] optimizer as described below:

$$v_t = \rho v_{t-1} + (1 - \rho) \cdot (\nabla_{\theta_t} f)^2 \quad (19)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t + \epsilon}} \cdot \nabla_{\theta_t} f \quad (20)$$

ρ defines the decaying sum. We compute the update but than divide the learning rate η for each weight by the new update. Thus oscillating weights will get a smaller update. Adam [2] optimizer works quite similar:

$$v_t = \beta_1 \cdot v_{t-1} - (1 - \beta_1) \cdot \nabla_{\theta_t} f \quad (21)$$

$$s_t = \beta_2 \cdot s_{t-1} - (1 - \beta_2) \cdot (\nabla_{\theta_t} f)^2 \quad (22)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{s_t + \epsilon}} \cdot v_t \quad (23)$$

We also adapt the learning rate per weight by dividing by the square-root of the squared gradients. But here also directly use the momentum but having the decaying sum of weight-wise gradients. β_1 and β_2 are tuneable hyperparameters.

1.5 LSTM theory

a) LSTM Gates

input modulation gate $g^{(t)}$ The input modulation gate determines candidate information from the new input (using also the old hidden state). We want our state values normalized but need also negative values (otherwise the cell values would only increase) which, as in this case, can be done with a tanh, squashing the input to $[-1, 1]$.

input gate $i^{(t)}$ The input regulates which and how much information of the input of this time step should be included in the cell and hidden state. As the input gate regulates the flow it is necessary to have its values bounded to $[0, 1]$ which can most directly achieved by squashing the values with the sigmoid.

forget gate $f^{(t)}$ The forget gate regulates which and how much information from the old cell state should be disregarded under the new information from the input (and the old hidden state). As the forget gate only changes the importance (magnitude) of the information in the cell state it should be in $[0, 1]$ which is achieved with the sigmoid.

output gate $o^{(t)}$ The output gate regulates which and how much information from the new cell state should go into the new hidden state. Again its gating the values from the other tensor which is asking for a range $[0, 1]$ achieved by the sigmoid.

b) Number of parameters

We have given $x \in \mathbb{R}^{T \times d}$ with T sequence length and d feature dimension. Further we have n hidden units. Then we have

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

trainable parameters for *one* LSTM cell. If we want to include the projection onto the classes c the size increases of course to:

$$4 \cdot (d \cdot n + n \cdot n + n) + n \cdot c + c$$

1.6 LSTM practice

Find the code inside `lstm.py` and `train.py`.

We train the two models (RNN and LSTM) for palindromes with sizes 5 up to 40. Both models get a hidden size of 128. We train with a learning rate of 0.001 until convergence of the loss. The weights for both models are initialized with He initialization [3] as this is more performs well compared to plain normal init.

For a overview of the results check out Figure 1. We see in general that the LSTM is able to learn the palindromes faster and for longer sequences. The RNN is only able to improve on randomness (accuracy of 0.1) up to length 17. Further the RNN is only able to reach full accuracy for palindromes smaller than 10. The LSTM learns for all lengths but we see that for length of 23 we do not see improvement over randomness until almost 3000 (if run for long enough the LSTM learns full accuracy for all tested lengths but we found that not be the interesting result here). This is explained in that we use the same hyperparameters for all experiments especially the learning rate. An optimization of the hyperparameters for the LSTM might speed up training for longer sequences. The experiment clearly shows that the LSTM is more capable of learning longer dependencies.

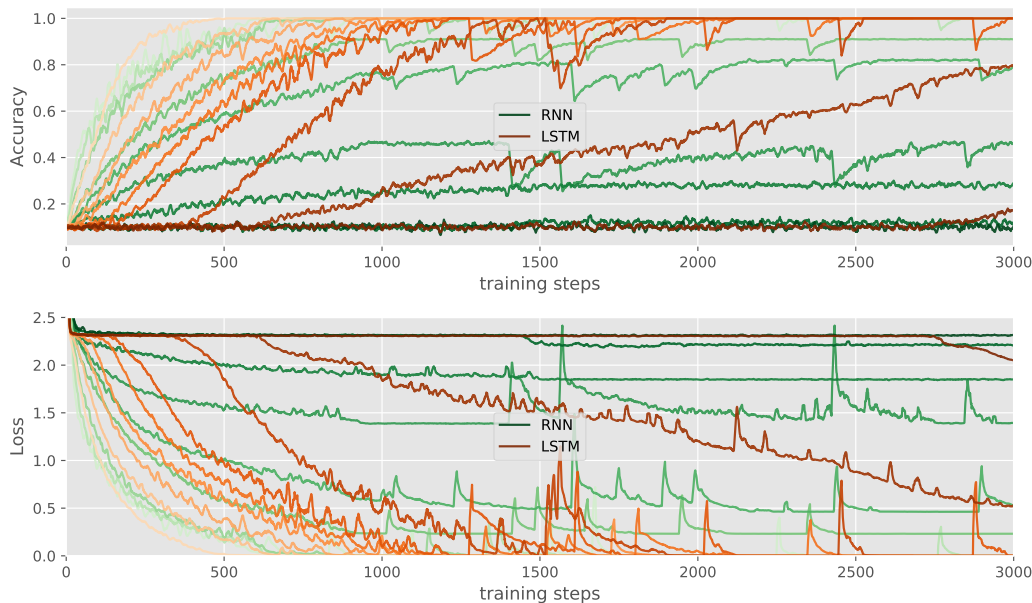


Figure 1: **Top** the accuracy and **bottom** the loss while training. Color lightness codes the palindrome length ranging from 5 numbers with the lightest color in steps of 2 to 23 numbers with the darkest color. (Making 9 curves per model). All curves are an average of ten runs and smoothed with a box filter with width 10 for better readability.

2 Recurrent Nets as Generative Model

2.1 Learning English from South Park and the Moses

a) The Implementation

Find the code inside `model.py` and `train.py`.

We implement the basic character-level language model using two stacked LSTM cells. Both cells are followed by a Dropout unit. We use a linear projection to project the hidden state from the second LSTM down on the class predictions. As we get a sequence of characters as input we use an embedding with fixed identity weights (one-hot encoding) to process the input for the LSTM. We set the hidden size of both cells to 128. The dropout layer is set at 0.05 dropout probability.

We train the model with the Adam optimizer feeding it sequences of 30 characters from the text in mini-batches of 64. The initial learning rate is set at $2e-3$. Further we decay the global learning rate every 5000 steps by 0.04.

We preprocess the text files by reducing the set of characters. All text is made lowercase and line breaks are removed. We remove special, uncommon characters (e.g. acutes from German and Spanish) replacing them: e.g. $\ddot{o} \rightarrow o$ and $;\rightarrow ,$.

b) The results

For training we feed the model with all transcripts of the television series **South Park**. In total the transcript have 813k words. See Figure 2 for the training progress in the first $1e5$ steps.

To understand the generative power of our model qualitatively we sample from the language model at different steps during training. Sampling is done by picking a first character uniformly by random. Next we iteratively feed the character into the model generate new hidden and cell states. From the hidden state we can sample a new character under different

temperatures. The new character is saved in the string and fed back to the model with the new hidden and cell state. We do this until we reached the desired output length.

Following are text samples of length 20, 30, 50 and 120 at step 0, 1e4, 2e4, 3e4, 4e4, 5e4, 1e6, 2e6 and 5e6. Each step we sample at temperatures $T \in 0, 0.5, 1$ and 2.0 . Note that temperatures 0 corresponds to greedy sampling, meaning we pick the character with the highest probability.

0			10000			20000		
Temp	1	Samples	Temp	1	Samples	Temp	1	Samples
0.0	20	riiiiiiiiiiiiiiiii	0.0	20	me to the stupid stu	0.0	20	9 the stupid people
	30	5iiiiiiiiiiiiiiii		30	re to the strange of the stupi		30	and the stupid asshole! wh
	50	iiiiiiiiiiiiiiiiiiii		50	that's not the power to the boys are go		50	g and the principal problem with the stu
	120	iiiiiiiiiiiiiiiiiiii		120	nna be a		120	dents and
0.5	20	oiiiiiiiiiiiiiiiiiiii	0.5	20	ze is the problem than the problem to th	0.5	20	can that the students and the students a
	30	iiiiiiiiiiiiiiiiiiii		30	e popped to the balls are gonna be so th		30	nd the police and the police and started
	50	iiiiiiiiiiiiiiiiiiii		50	e problem that the problem of the part o		50	to the police and state the part of the
	120	iiiiiiiiiiiiiiiiiiii		120			120	
1.0	20	yjw2b3b6 nd. w,m54tw	1.0	20	! i know i thought i	1.0	20	-hold on. god damni
	30	7-edbx2)dg4n7syam enlahvã sx-		30	11 and sooo and then we can be		30) that's what i don't have a
	50	r8'/du1c6qzclsi7.lkismwoui"xznã68-a8gg z		50	got a spilitory. the hell are you beauti		50	ou want to show the poor more than your
	120	w'e)!/pb6		120	ful his mo		120	problem st
2.0	20	kee,e),j8noza,a/6enra""h 'f'a79d'0gãu'ya	2.0	20	/fuck your son in your new problem and i	2.0	20	5 from here for here! we can do it for
	30	!nr-(opt,s(t ode)!..hz3z9(sc'jq/b zpr5ãc		30	'm sure your looks like the lately his n		30	a boy is for a bad of the planet. what a
	50	txlm9uid(bnd9zr6/.p ãlñpeglm nzxh ?7nksx		50	ight to the bame! and we have to play wi		50	re we gonna do it out! what are we gonna
	120			120			120	
3.0	20	e!28jn25(1815yzzwthr3	3.0	20	nrish are dellagin's	3.0	20	tle son is so any ne
	30	0..fg'(iaa 9'ããã)d6a 81i7oabaf		30	hiltor. alto si-.ããdisolant s		30	quut up and shows! we want it?
	50	ptnctnng,h k.-6ybilb1/.mdf5a(8xenxmmu7bsx		50	quacking are on mandayout kike the burg.		50	fell is whatever if here in the stuff th
	120	du3lvt95b9		120	that's? k		120	at we coul
4.0	20	rx54 ?5taif'cs1c)la36wsy0'2eodj6/?y.5yq?	4.0	20	lor place! all what we've reading much-b	4.0	20	! grabba basky with a news. no, kyle. ma
	30	q/2 ocy5,(ctzq'ln?sojh38ahgp() 9m3.p9lhs		30	leh'is is broolly hangh! keep you! i'm ju		30	ybe you talk to you that? i don't know y
	50	7918h3lã1ãf1"0.n3s13ve zk'ã,u85(tt dp		50	st gonna kill by a hot. .r. tom. yeah! h		50	our veal. my pockets, zammate your legra
	120			120			120	
5.0	20	ar8ld7gb7n!s5pzxj 5f	5.0	20	yed. yoh,! whaaa, wha	5.0	20	8 totarnap quiete ch
	30	ufnurr,fnc2o)h9hy?9.fkpn8)8wv		30	, huh8mu. nsy! oh what!,'yeaw		30	(clein! lond.ncomaa-lopeous c
	50	jkw(-unz1vm'n3'ã/?r6rz(le0ã c8q k0(pnff		50	yetriuze be bocy toks. ugs! 4" - you ju		50	p.,"" nice, nobody! g'ee!! yemph.) o-how
	120	4.oy7)san3		120	st lose wi		120	seefo elea
6.0	20	?-pi!'8l6)n-qj/50.j.naq7'ebjvnm!ã"c02mj	6.0	20	(122.ã peofjies, bervltsalyfatquemasine,	6.0	20	2!...lu-yuwwakkih? caught. ""n'tcufort, d
	30	,0t9 vekhj-7zbw-8f6j-dzq?-m,(q7z"73s)7-u		30	you do've fwire 69fven cull,! 'bye.!!)!		30	nivo wegghy newra! geraldon(ensides you
	50	rn eqtã"nd2omjji8 (7j30xylc9ore.ubon762i		50	me! it's ville,""days, sv!!l poft one		50	know! in-carv twa! (dudd-k egerzipzsnc.
	120			120			120	

30000			40000			50000		
Temp	1	Samples	Temp	1	Samples	Temp	1	Samples
0.0	20	per the problems and	0.0	20	the stupid ass of t	0.0	20	- and then i can se
	30	e the problem is a little boys		30	the stuff in the students and		30	f the students are so that the
	50	n the protect the probably in the contro		50	you guys are gonna be a children and sta		50	ing to see you all the start on the part
	120	l of the p		120	y out of h		120	y of the p
0.5	20	- and i don't know what i was the paren	0.5	20	s and the problem that the state of the	0.5	20	re a little boy in the principal stupid
	30	ts are the stuff and i was the problem t		30	students and the problem that the studen		30	asshole! what the hell are you doing? i
	50	han the park country in the state to the		50	ts and start a second of the stupid ass		50	don't know what i can do it all the stu
	120			120			120	
1.0	20	? oh my god! well,	1.0	20	ll you have a drink	1.0	20	good, the only perso
	30	g all the kids hate your vampi		30	t that that we got reality who		30	10 people who present. you don
	50	/friends to put the bus. well, sure is a		50	t his son's friends and a second. you gu		50	2 mm. cartman, you're not going to make
	120	little fu		120	ys take a		120	them bette
2.0	20	day cartman! it sure is true. he's kille	2.0	20	call my house. that thing is stoppin' he	2.0	20	s on the people be a little boy the stat
	30	d, and i want to do is what you understa		30	r canadian for a time! ...and i know ho		30	us of her there. tell you in here. i can
	50	nd that there was my friends and some ho		50	w you wanna do this and well then i have		50	't take it anymore was some construction
	120			120			120	
3.0	20	me act now. that's w	3.0	20	ãdreambit careful, i	3.0	20	okay. 'cause you com
	30	give me, even going up with my		30	xice how me a mall eating. tha		30	ll i've... good way, and sweet
	50	2s? i don't know a people have a lot of		50	ball maybe insured on our odmedent. it'		50	7 friends off, too. my legs of force her
	120	words agai		120	s not on t		120	e he give
4.0	20	5 down't you guys are not all the show t	4.0	20	aking a bunch about you! alright. oh, c	4.0	20	pash! you decighcion the poxauld first
	30	ogety goddamned doctor is a crow, butter		30	ows! that's nice back that does. and uh		30	comes in trualing that saying this time.
	50	s! that does with ana good oraig, where'		50	yeah, it'll get the titzolisy. six stor		50	oh yeah. okay kids, donleya! ned genefi
	120			120			120	
5.0	20	n aypofa? yeah!!) u	5.0	20	-e! your habsy. eric	5.0	20	u don't know. ãriva
	30	quedsplode., not theared lovin		30	7395.""- " stapa.nlona ded, you		30	julth not an/mh??pkmman! holy,
	50	yw. xlezh, v mrtol! govertuh you ya! um r		50	iaand quest monmils, eric. moutupt a!!! t		50	tyall! we!sex, subbeld's bad alright-..i
	120	test. hyeh		120	hanks, cla		120	mul! bemp
6.0	20	6e-c6? ma'poghutlike,"waitbmecy. a adulc	6.0	20	ah'-'! ..ohrs, who duid "c?ãcalmndy!) ta	6.0	20	?,-ããuh i camebokalo's?!iravy?! oh, s
	30	cuma"" zey ninety, 3.6 wqi-fs' woman?!ãk		30	uh tox into ys-ave. shho! o-kay! you,		30	top!-uw, all, uuuf-if there suddes8 her?
	50	yre? thh). wamnasiqn! 4) i-cake it gly		50	9, quickly. (kittle koek!!r me cy were		50	unfniyor mado. start! afriewt! your l
	120			120			120	

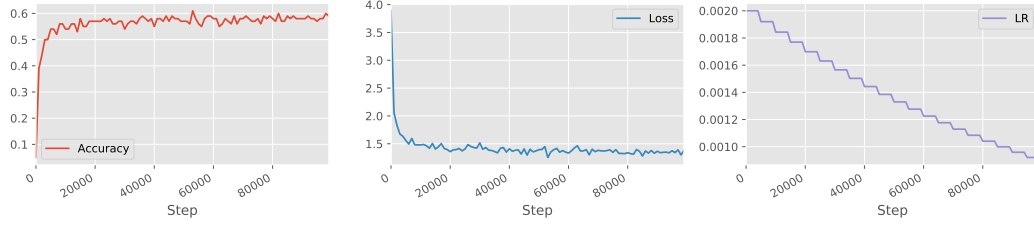


Figure 2: Accuracy, Loss and the decaying Learning Rate for the LSTM language model during training using the South Path transcripts. Note that accuracy and loss are a bad surrogate for our task as they describe the capability of the model to predict the next character given a string while our actual goal is to get a generative model.

100000			200000			500000		
Temp	l	Samples	Temp	l	Samples	Temp	l	Samples
0.0	20	f the stupid asshole	0.0	20	xt to the bathroom a	0.0	20	x and then i can see
	30	we have to get the star really		30	40000 per cent of the state of		30	, i want to be a company that
	50	u guys are the only thing to do it there		50	(a friend is a little bit of a bitch! w		50	quite a lot of money to see the start a
	120	and i don		120	hat the he		120	lot of mor
0.5	20	80 per cent to the soul of the state and	0.5	20	n the stupid party of the store for the	0.5	20	5 month. what are you doing? we have to
	30	so the school stay away from the countr		30	second that the world is a little boy wh		30	do it the students are the one that the
	50	y is a big deal with the start a second		50	o was the problem. what are you doing?		50	world is a little bit of the problem.
	120			120			120	
1.0	20	ke the pain! we want	1.0	20	85 dollars signing t	1.0	20	n i don't feel so m
	30	2000, i'm a big and i can't do		30	ing the town of a super be sat		30	3 dollars. i love you guys, i
	50	ou guys! yeah, we have to work at my bod		50	00 per cent of them. hello eric, that's		50	/11 has pretty good to go one different
	120	y. i don't		120	it there.		120	boys and f
2.0	20	we have to kill you six hours. they did	2.0	20	ke in a son is about to be a big deal to	2.0	20	2002! he was the united states with the
	30	it! the things to get a really seeing t		30	god is having a new town of all the red		30	armas place in the fish and that's okay.
	50	he one things is. we are the brains and		50	life in sandwich thing? so for you to s		50	all right, we are so funny! we can do i
	120			120			120	
1.0	20	ht at us saw... all	1.0	20	for that idea. are	1.0	20	w, that noticed you!
	30	? don't they come a meters! wh		30) doh, if a scott. i know we'		30	- she's too... ruuuurribuy! t
	50	, you know what they aren't any of that		50) your sriss of mouths of charge of you		50	10,000 one to this out of hero...you'll
	120	guy travel		120	de by him		120	meet a daw
2.0	20	? really? you have to be costaiint. so,	2.0	20	r you now bitching my part of my towelie	2.0	20	gonna build your flute boy in my olved q
	30	nuh dougless is the confederated chinpo		30	!! i know it here? look, we will cut off		30	uit. go sure. it's serious! every piss,
	50	kes you dope? could you land it... sandw		50	my balls, red measure?! get your grandm		50	huh?! oh, hey, changedage the stupid au
	120			120			120	
1.0	20	, ver disk thos girl	1.0	20	goaj. (go we'll bo,	1.0	20).qh avoid theng "h
	30	(k. graswmak!! hs, ly- wen'- n		30	evweac eated itge-joren l.? i		30	- ad zumir's reevc dlamant! l
	50	4 ! eric cat, kyloh, -mosely! -'sssuper		50	've gravi my vacco., "s?? kewnu , ms. ng		50	ub! yeephclexto, cartman, the run. o,...
	120	ing,s waal		120	-ctdai.,mi		120) boy. ste
2.0	20	? bopha nicpdorations, a chinaple th you	2.0	20	872!0 yuk. ?que? whe- hey mirgith rouetn	2.0	20	- get mm."" nribly dude! p-juros.!!fiz
	30	r-rect kickin'cybun tripi and 'cd'of-)		30	put yoba goddami-engal"" go back with 3/		30	e i bb gallawes-c-curity!"" caren sog
	50	whh ibz. yem.y?) oh... with?. (hm.!) if		50	5. obceaob, ike, 'f's lots autlakffful i		50	rth! right ninleyts, garyeow? i'm i?" ...
	120			120			120	

We see some general trends. At temperature of 2.0 the model never able to form words which suggests that this is too high. Most interesting we see the model being able to generate almost coherent sentence early on in training as e.g. at step 1000 with $T = 0.5$.

In a second experiment we feed the network the **five books of Moses** from the old testament. Compared to the previous transcript this text only has 156k words. We extend the model architecture to check if that increases the generative power. For that we us *three* stacked LSTM cells. Further we feed sequences of 40 characters and decay the global learning rate only all 20k steps. All other parameters are set as before.

Trainings progress was similar to the first experiment so we just give exemplary qualitative results below.

3 Graph Neural Networks

3.1 Forward Layer

a)

The \hat{A} matrix contains the edge information between the nodes in the graph. This includes the self-connections of all nodes from the identity $\mathbb{1}_N$. We update the activations with $H^{l+1} = \sigma(\hat{A}H^{(l)}W^{(l)})$. So we update the activation associated with one node only for the

activations of edges that have non-zero edge weights in \hat{A} . A node is changed by the nodes its connected to. In reverse we can see that information in one node can propagate to all connected adjacent nodes in one time step which can be visualized as the message passing over the graph.

b)

In every layer of the GCN you can propagate information along one edge so to let information reach a node three hops away we would need three layers.

3.2 Applications of GNN

In *Multi-Granularity Reasoning for Social Relation Recognition from Images* [4] the authors build graphs on images of humans. The graphs describe first the relationship of a person in the image with its surrounding objects amongst other things other persons in the image. A second graph represents the pose of each person in the image. On these two types of graphs they use Graph Convolutional Networks (GCN) to predict the social relationship of these persons. For example in an image of parent and child the bending pose of the parent and the connection of the two persons in the Person-Object graph lets the method infer their relationship.

Next in *Disease Prediction using Graph Convolutional Networks: Application to Autism Spectrum Disorder and Alzheimer's Disease* [5] propose to use GCN for medical image processing. The nodes of the graph in this setting are features of medical image acquisitions which in their experiments are gathered from structural and functional MRI. As they predict the health state of multiple individuals at once the graph consists of many image features of multiple persons. The edges here describe the phenotypical similarity between two individuals which are described by categorical medical data (e.g. sex). The GCN uses this graph to predict the health state of the population.

Lastly in *Temporal Relational Ranking for Stock Prediction* [6] we seen an application of GCN in stock prediction. Here the nodes are features capturing the historical information of one stock and the edges capture the relations between two companies stock. Both historical intra and inter stock features are generated by LSTM from historical stock data. Again we build a graph from the edges and nodes and use GCN to predict in this case the stocks next day performance at the market.

3.3 Comparison and Combination of GNN and RNN

a)

Using a RNN assumes our data can be is orderable along one axis meaning every node of information is at most connected to one *previous* and one *next* node of information and that no two nodes have the same previous or next node. For the GNN we assume data whose node of information is related to an irregular number of other nodes.

- given relationships weight

b)

We saw such a combination of RNN and GNN models in Section 3.2 already. The work on prediction of stock markets used LSTM and GCN in conjunction. In there each node of graph related to a linear temporal structure, the historic sock data. As those linear temporal graphs where themselves related to each other one can use a graph model to analyze. The reverse idea also seems plausible by using a RNN to go over nodes of graphs. Imaginable might here be an application where the data is organized in a graph, let's say a social graph, but is subject to temporal changes. A GNN could be used to analyze the state of the graph at one given point while the GNN uses these temporal states and can for example predict future states from that.

References

- [1] Geoffrey Hinton. Lecture 6 Overview of mini-batch gradient descent. URL http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.
- [2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization.
- [3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. URL <http://arxiv.org/abs/1502.01852>.
- [4] Meng Zhang, Xinchun Liu, Wu Liu, Anfu Zhou, Huadong Ma, and Tao Mei. Multi-Granularity Reasoning for Social Relation Recognition from Images. URL <http://arxiv.org/abs/1901.03067>.
- [5] Sarah Parisot, Sofia Ira Ktena, Enzo Ferrante, Matthew Lee, Ricardo Guerrero, Ben Glocker, and Daniel Rueckert. Disease Prediction using Graph Convolutional Networks: Application to Autism Spectrum Disorder and Alzheimer's Disease. 48:117–130. ISSN 13618415. doi: 10.1016/j.media.2018.06.001. URL <http://arxiv.org/abs/1806.01738>.
- [6] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. Temporal Relational Ranking for Stock Prediction. 37(2):27:1–27:30. ISSN 1046-8188. doi: 10.1145/3309547. URL <http://doi.acm.org/10.1145/3309547>.