LSTM Language Model: Text Generation

Anthony Tri Phap Nguyen
Kungliga Tekniska Högskolan
atpng@kth.se

Christoffer Ottersten
Kungliga Tekniska Högskolan
chrott@kth.se

Abstract

In this paper a set of language models (LMs), based on recursive neural networks (RNN) laced with long short-term memory (LSTM) layers, are trained and evaluated by utilizing the generative capabilities to predict text. The quality and coherency of the language from the generated texts is investigated for each model and compared with each other and against a set of base references. In conclusion, the results from our experimental procedure revealed that the least complicated RNN (only one LSTM layer) achieved the best text generation performance.

1 Introduction

Automatic speech recognition (ASR) is a field within computational linguistics that has extensively been researched throughout decades due to its ability to create speech driven user interfaces. This has several applications in areas including medicine, military and in-car systems. ASR converts natural speech into a sequence of words through different means of computer algorithms. Language models (LMs) is an important feature addition to ASR as it has shown to improve performance of speech recognition [4]. The fundamental idea of LMs revolves around the ability to impose language restrictions into the system in order to assist in minimizing recognition errors. In this study we propose to train and evaluate the performance of neural network based LMs through intrinsic evaluation and by qualitatively analysing text generated samples from the trained models.

Traditionally, n-grams have been the most frequently used LMs, which is a statistical LM that relies on evaluating probability distributions over word sentences to minimize recognition error. Their simplicity and proven efficiency is one of the reasons they still tend to be the most widely used LM [12, 9]. However, the disadvantage of n-grams is the need of large sets of training data whilst inadequately capturing long term dependencies in natural language [4].

Recurrent neural networks (RNN) based LMs have been proposed to provide a solution to the long term dependency problem as well as achieving similar or improved performance by utilizing smaller sets of training data [7, 8]. Through a sequence of words or characters the neural network LM is able to process and predict the most likely outcome. The main structure of a RNN is very similar to that of the feed forward neural network. An input is fed to hidden nodes where the information undergoes computational treatment before generating an output. What makes RNNs unique is that the output is additionally re-fed into the node to be used for the next input. This process is beneficial for sequential data where it is preferred to assume that the information is connected and not independent (feed forward networks) [7, 8].

Furthermore, studies have shown that a specific type of RNN incorporated with a long short-term memory (LSTM) performs even better with regard to long term dependencies [1]. As an extension of LSTMs the general network structure is identical, however, with additional features in the hidden state that allows for appropriate information handling

LSTMs have been proven to be extremely useful and are currently used in a variety of areas such as speech recognition, machine translation and language modelling as well as possessing generative capabilities [10, 13]. Research of LSTM LMs have shown to outperform state-of-the-art n-gram

models trained with 100 times more data [4]. Trained LSTMs have been used to generate music,html pages, text and even scientific reports (although nonsensical)[5, 11, 7]. Conversation based models with LSTMs have shown that coherent generated responses can be achieved [11]. It is hence assumed that a good LSTM LM should generate coherent text with long-term dependencies. LSTM LMs are evaluated either intrinsically (calculating perplexity) or extrinsically (implementation into a speech recognition system). Studies tend to evaluate LMs both ways, although there is consensus that extrinsic methods provide a more reliable evaluation of the model, however, it is a cumbersome process compared with the cheaper and more efficient intrinsic methods.

As the research studied all tend toward using different LSTM architectures for language models, conversation based models and other applications we propose to examine a variety of architectures to draw conclusions on their relative performance. In this report we investigate the performance of several LSTM based LMs by intrinsic evaluation as well as evaluation of text generated samples from each model. In section 2 we discuss the details of the methods used in order to investigate and evaluate different LSTM LMs. In section 3 the experimental set-up with details about text generation is discussed. The report concludes with a presentation of results followed by a discussion and conclusion of said results.

2 Methods

2.1 Neural Network Architecture

The basis of our study is to investigate LSTM LMs, thus a more extensive analysis is made into the general structure of RNNs and LSTMs in order to grasp the variety of parameters and architectures that can be used. The structure of a RNN is easily visualised in Figure 1, where information from previous time steps are utilised in the current time step and thus preserving dependencies in text. The input of RNNs are vectors which undergo a transformation by weight matrices to generate an output vector. The weight matrices are constantly adjusted throughout the learning period of the network by a back propagation algorithm. However, as RNN are deep neural networks the gradients calculated from the back propagation algorithm exponentially tends toward 0 hindering long term dependencies to be learned, known as the vanishing gradient problem [1].

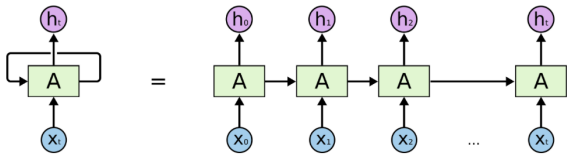


Figure 1: Two visualisations of the same RNN network [1]

LSTMs have been proposed to surpass the vanishing gradient problem and therefore has the capability of learning long term dependencies [13, 11]. This is achieved through added data handling layers in a LSTM unit (cell) as seen in Figure 2. The most important aspect of a LSTM cell is the cell state, which essentially contains the information of each cell. Additionally to the cell state there are three cell *gates*; *input gate*, *output gate* and *forget gate*. Each of these control what information is to be stored, passed on and forgotten for each cell state and are governed by sigmoid functions, as seen in the Figure. Weights are applied to all *gates* and are adjusted throughout network training.

There are several variations of the LSTM cell that have been used throughout research to improve performance [1], however, we focus on alternating the amount of hidden nodes and LSTM layers instead of alternating the functions of inside the LSTM cell. Figure 3 is a crude visualisation of the basic idea of multiple layered LSTMs.

2.2 Neural Network Training

In order to create adequate LSTM LMs, the network needs to be trained appropriately, through processing a text set (known as a test corpus). The input to the LSTMs are words which are represented in vector form, the output is the most likely predicted word given the input.

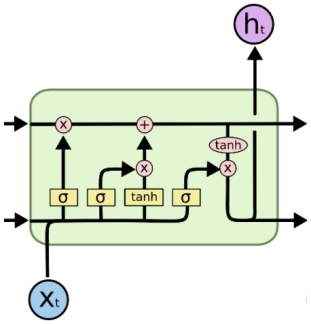


Figure 2: Detailed insight of a LSTM cell [1]

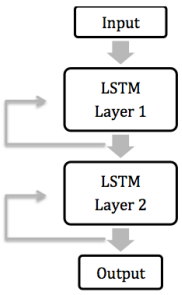


Figure 3: Multiple layered LSTM

Neural network training is performed using a text corpus, this is essentially a plain text file. It is necessary to sub divide the text corpus into a training and validation set. As the model is trained it sequentially calculates the loss (both for the training and validation set), using a specific loss function, of the system and thereafter readjusts the weight parameters and repeats this process until the loss of the system converges. The output of the model is dependent on the input corpus, it is therefore necessary to use "clean" text without abbreviations and odd symbols, unless wanted. The context of the text trained is mirrored in the output [5, 7] and it is therefore important whilst evaluating the LM to take this into account.

2.3 Text Generation

LSTM LMs predict an outcome dependent on a specific input sequence. In order to generate an adequate amount of text from an LSTM LM the output is fed into the network in a repeated fashion until a text sequence of desired length is achieved.

2.4 Performance and Evaluation

Evaluation of language models is either evaluated intrinsically or extrinsically. As stated in the introduction different studies use either one or both methods when evaluating. We propose to evaluate the LSTM LMs intrinsically, through evaluating the loss throughout training of the models as well as generating text for each model and evaluating against a reference sentence and against each other. The loss calculated is directly correlated to the per-word perplexity (perplexity) of the model according to the formula below.

$$Perplexity = e^{loss} \tag{1}$$

Hence, low loss generates low perplexity which points towards a better LM. However, this is a rather crude evaluation and one can only generate reliable evaluations by implementing the LMs into an

ASR and evaluate its performance. There is, however, research that suggests a correlation between lower perplexity and increased performance through implementation [9].

An important aspect with LMs is the ability to evaluate the quality and ability to preserve long term dependencies. We propose to do this by generating text (through a specific input sequence) and evaluating its performance based on specific reference text, overall coherency of the text and relative performance.

3 Experiments

3.1 Implementation

For the RNN implementation we will make use of the deep learning library *TFLearn*, which is build on top of the open source library *TensorFlow*. The library has build-in functions to build a RNN and add an arbitrary amount of LSTM layers. The TFLearn model will then be connected to a sequence generator which will generate a specified amount of characters based on a given test sample.

3.2 Data Preparation

All test experiments are conducted using a Celtic tale subset from the Fairy Tale corpus [2] consisting of `34551` words in the training set and `3839` words in the validation set. The simplicity of language, appropriate length and lack of special characters or abbreviations makes it an attractive corpus to use for training. Due to time constraints, the text corpus chosen is small compared to other studies that perform training with text corpora of several million words [13, 10, 4].
To test our trained networks we will feed part of a different fairy tale into the RNN and let the model sequentially generate characters and words learned during training. The networks will be forced to generate a short continuation of the story for `500` new characters.

3.3 Neural Network Parameters

For our study case we will be testing four different network layouts:

1. RNN with one LSTM layer containing `512` units
2. RNN with three LSTM layers containing `512` units
3. RNN with three LSTM layers containing `1024` units
4. RNN with three LSTM layers containing `256` units

In the first two cases we want to investigate the improvement in performance when adding more layers into the network. The third case will also increase the network complexity by increasing the number of units within the LSTM layers, which will also increase the risk of over-fitting to the training data. As contrast we will train a network with less LSTM units per layer than the second network. Depending on the dataset, this could increase the generalisation of the RNN, in case `512` units were already enough to capture most of the dataset.

3.4 Text Generation and Performance Evaluation

The evaluation of the models is mainly based on the different texts generated from the language models. In order to evaluate the models equally, the same initial input is used to generate the text sequences of the models. Two input sequences where used to generate text and are displayed below, they correspond to sentences extracted from the training data and externally sourced Celtic fairy tales. The generated texts are then evaluated against each other and how well they correspond to the actual continuation of the input sentences. In addition a crude evaluation metric for the percentage of incorrectly generated words is provided. Furthermore, the generation tool used for this scenario has an adjustable *temperature* parameter, which influence the "creativity" of the model. The parameter can be set between `0.0` and `2.0`. Setting it to zero will tell the model to strictly only use words/character sequences that it has learned during training, while a high value loosens this restriction.

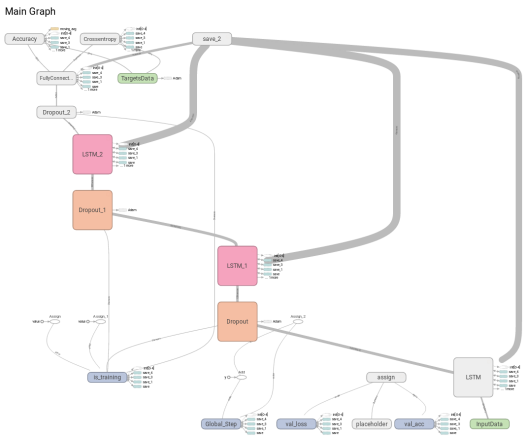


Figure 4: Network model of RNN Nr.2

- Training set input seed: *"That evening, too, Princess Rose went out on her balcony and clapped her hands. But when the golden bird ap"*
- Training set sentence continuation: *"...peared, her hair glowed in black instead of red. The bird warbled its enchanting melody, and Princess Rose sang her lullaby."*
- Test set input seed: *"In olden days, when many Kings reigned throughout the Green Island of Erin, none was greater than the great"*
- Test set sentence continuation: *"...Concobar. So fair was his realm that poets sang its beauty, and such the wonder of his palace that the sweetest songs of Erin were of its loveliness."*

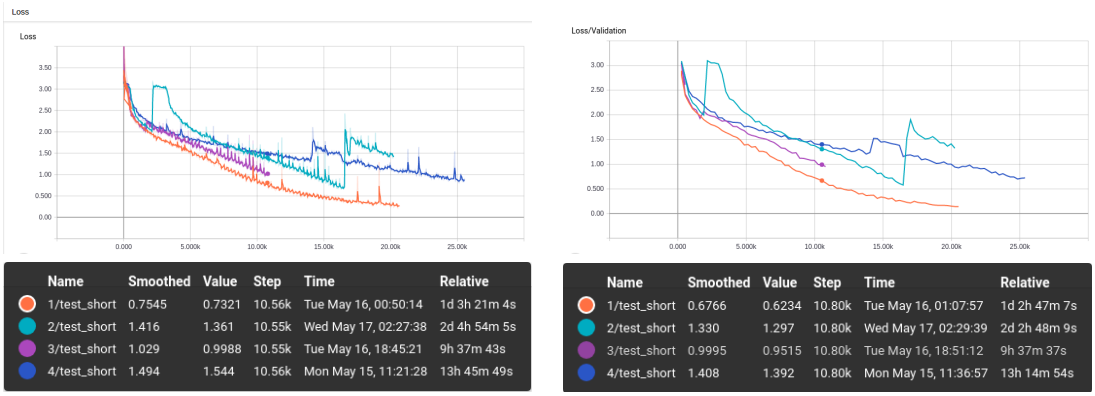
Given the input to the different LSTM LMs, we investigate the similarity between the reference text and the generated text. The loss function is also a used as an analysis tool and is calculated by using the categorical cross-entropy on the probabilities determined with the Softmax activation function. To reduce the cost, we use the Adam optimiser and a learning rate of `0.001`. A low loss value is a indication of a better LM, however, this does not necessarily mean it performs well once implemented into an ASR. Although, ideally, the aim is to be able to compare with a reference sequence, there is a very high probability that the generated sequence might be nonsensical.

4 Results

The following figure 5 displays the training and validation loss of the four neural networks during training. The training has been conducted for different amounts of training steps as the training duration varied for every network but to compare the networks we will evaluate them in a state when they each have passed an equal amount of training steps as indicated in the figures. The `Relative` value in the legend window cannot be taken as reference for the training duration, as the networks have been trained on processors with different computational speed.
It might be a more reasonable approach to compare the networks after they have converged. But due to time constraints, this approach might not be feasible, especially after observing the calculated loss functions. While the third RNN has a continuously decreasing loss, the first network shows tendencies towards convergence for the duration of our training. The forth and particularly the second RNN show sudden jumps to a higher loss value from which the network will have to re-train again. An additional training from scratch of the second network has been performed to confirm if this was an actual behaviour of the network and not from our experiment set-up and the resulting loss function had a similar progression but with different amplitude (figure 6).

4.1 Using test sample from training set [2]

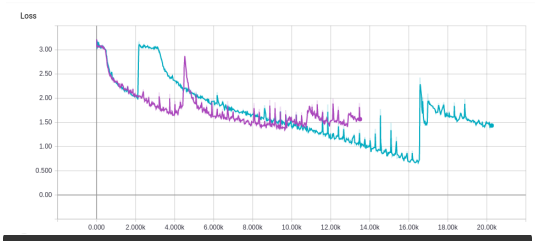
In this section we used a test phrase from the training set and test set as a input seeds to evaluate the network performance on its ability to generate coherent text with the same context. The



Name	Smoothed	Value	Step	Time	Relative
1/test_short	0.7545	0.7321	10.56k	Tue May 16, 00:50:14	1d 3h 21m 4s
2/test_short	1.416	1.361	10.55k	Wed May 17, 02:27:38	2d 4h 54m 5s
3/test_short	1.029	0.9988	10.55k	Tue May 16, 18:45:21	9h 37m 43s
4/test_short	1.494	1.544	10.56k	Mon May 15, 11:21:28	13h 45m 49s

Name	Smoothed	Value	Step	Time	Relative
1/test_short	0.6766	0.6234	10.80k	Tue May 16, 01:07:57	1d 2h 47m 7s
2/test_short	1.330	1.297	10.80k	Wed May 17, 02:29:39	2d 2h 48m 9s
3/test_short	0.9995	0.9515	10.80k	Tue May 16, 18:51:12	9h 37m 37s
4/test_short	1.408	1.392	10.80k	Mon May 15, 11:36:57	13h 14m 54s

Figure 5: Training & validation loss of four different RNN



Name	Smoothed	Value	Step	Time	Relative
2-2/test_short	1.546	1.529	13.31k	Wed May 17, 08:31:27	2h 48m 15s
2/test_short	1.069	1.065	13.31k	Wed May 17, 03:01:01	2d 5h 27m 27s

Figure 6: Training loss of second RNN

`temperature` parameter has been set to `0.5` and `0.1`. For our evaluation, only the generated text from the first and third network is included in this section but the text output of all four networks on the test set [3] can be found in the appendix of this report. Table 1 and 2 demonstrate the percentage of incorrectly generated words for the different networks for the training and test input seeds.

In general, the simplest RNN1 performs the best in terms of generating valid words, as is portrayed in Table 1 and 2. The difference of setting the temperature to `0.1` and `0.5` is not very noticeable - about half of each sequence are valid words, but the overall meaning is equally worse.
The third network shows a much worse performance. While almost all generated words are non-valid, setting the temperature to `0.1` will make the network generate only blank characters after awhile.

When using a test sample from the test set and `0.5` and `1.0` for the temperature (see appendix), a more distinct difference from the choice of temperature value is observable. More valid words can be found for `0.5` except in RNN3, which has trouble generating words in both cases. For the latter network the generator has a tendency to longer words with higher temperature. RNN4 even starts to repeat most of its used words for temperature `0.5`, which can be an indicator of lack of network complexity and low memory capacity of the network.

Judging in terms of valid syntax, RNN1 arguably takes the lead again. From the text generation using the test set as input (appendix), said network shows a correct positioning of subject, verb, etc. multiple times. Overall, all four networks have great difficulties at generating valid text with meaningful content in our experimental set-up.

4.1.1 RNN1: one LSTM layer, 512 hidden units

```
- Test with temperature of 0.5 -
In olden days, when many Kings reigned throughout the Green Island of Erin, none
was greater than the great goor for thee, not the Wine of Finn who paast bich diely
chee.  King here in whice thee hore of a walred tiee, with head hrmmed until eyen
from the wind sood, and eye were the given then to her thice the brither wasering to
Finn's heart, for when the chief who lave on which hur him the forge fave he playe
surling surlings wife her him follow.

And deamies was not bear live and sweet, for when I sally were asled of the wind
would nett hie surre.  The was the forest, and at is foot behild you fort

- Test with temperature of 0.1 -
In olden days, when many Kings reigned throughout the Green Island of Erin, none was
greater than the great go but quinca, and with them the children of Lir reached the
words of the plain before the Green Islen would nother with magh room he spely the
wandord.  But ere the three hundred mad brought with him there.  Thee forth ho sear
upon his broed by the hould.

Then dad Finn shall his chiefs, and the forest and was sorrow all the body of Finn.
And the fore of the magic aslin be the sen of Meyle.  Then Ass and sare I rowe as
the forest, and of ir ass the came of the Soar?

'I lave the goars came rowhe
```

4.1.2 RNN3: three LSTM layers, 1024 hidden units each

```
- Test with temperature of 0.5 -
In olden days, when many Kings reigned throughout the Green Island of Erin, none
was greater than the great fe f r h m encehed ne our hee tno ye,t thn oreeeleh r
thoai wou wai the Kea saaatei anl oue fro en eet r ng h m two haaahe th s thsa shone
nhelle the o n ald d e facithed tha hite a ehery tdere felh tleare.

fu hewe ma i ne e dA whit r t erand ted raise fot thete fr with and the a withs Br
f unne neopy tnhearah cam ito the c steen f t n th sh th de, an t,ne wouod.  A lone
myr no maide with maou.e sd lime.  h ne waar at t Bear ne what fair to ou

- Test with temperature of 0.1 -
In olden days, when many Kings reigned throughout the Green Island of Erin, none was
greater than the great fe w th t Fi lt the bo t A m tho e
```

Table 1: Incorrect word generation rate for test sample as seed

Neural Network	Word Error (Temperature *1.0*) in %	Word Error (Temperature *0.5*) in %
1	29	25
2	52	33
3	~ 100	~ 100
4	32	25

Table 2: Incorrect word generation rate for training sample as seed

Neural Network	Word Error (Temperature *0.5*) in %	Word Error (Temperature *0.1*) in %
1	29	18
3	~ 100	~ 100

5 Discussion & Conclusion

For the relatively small dataset we used in our experiment, our simplest recurrent neural network with one LSTM layer had the highest performance. Both its training and validation cost function reached the smallest value when comparing the networks after equal amounts of training steps, but in the end none of the networks were able to recreate a context-related text from the training set. Although it was originally planned to compare the generated text with a reference text by using a metric evaluation system (WER), it was noticed that the models weren't capable of generating complete text and a crude metric evaluation system based on incorrectly generated words was used. Although it demonstrates some interesting percentages it neglects any form of sentence structure and therefore, a qualitative analysis was more valuable. The first network was closest to generating text in a valid syntax, while also generating the largest amount of valid words from all tested networks. We therefore conclude that the first neural network based LM performed best in terms of its capability to generate somewhat coherent text. Another interesting fact is that certain long term dependencies can be seen in the text, such as names of characters associated with the seeded input text.

Neural networks 2 and 4 exhibit sudden jumps in the loss functions during training. These could occur from an unstable state of the network were it experiences an error it tries to compensate, but to the recursive nature of the network, the error exponentially grows until the error naturally drops out or the network manages to re-stabilise. Nonetheless, further research has to be done on this topic, using a different dataset with a different size and training the networks for a longer duration to see if the jumps occur regularly and whether they will have an influence on the convergence capabilities of the networks.

A continued study can also be conducted for the generality and word prediction capabilities of the networks after training them on a much larger dataset. It is also possible to include a metric performance system such as WER (word error rate) once complete text is generated. To speed up the training duration, the learning rate can be increased. Alternatively, reducing the impact of the decay rates `beta1` and `beta2` in the Adam optimiser by using a value closer to one can also help with the time problem.

References

[1] C.Olah `http://colah.github.io/posts/2015-08-Understanding-LSTMs/`

[2] Celtic Tales from Fairy tales corpus `https://github.com/bgmartins/fairytales-corpus`

[3] Test sample for text generation `http://www.worldoftales.com/Users_stories/Princess_Rose_and_the_Golden_Bird.html` (7th paragraph)

[4] M. Sundermeyer, H. Ney & R. Schlüter. From Feedforward to Recurrent LSTM Neural Networks for Language Modeling, *IEEE/ACM Transactions on Audio, Speech, and Language Processing 23(3):517-529*, 2015.

[5] A. Graves. Generating Sequences With Recurrent Neural Networks, *Computing Research Repository, 1308.0850*, 2013.

[6] Y. Bengio, R. Ducharme, P. Vincent & C.Jauvin A Neural Probabilistic Language Model,*Journal of Machine Learning Research 3:1137-1155*, 2003.

[7] I.Sutskever, J.Martens & G.Hinton. Generating text with recurrent neural networks. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011.

[8] T.Mikolov, M.Karafiát, L.Burget, J.Cernocký & S.Khudanpur. Recurrent neural network based language model. *Interspeech 2:3*, 2010.

[9] T.Mikolov, M.Karafiát, L.Burget, J.Cernocký & S.Khudanpur. Extensions of recurrent neural network language model. *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference 5528-5531*, 2011.

[10] W.Shuohang & J.Jiang. Learning natural language inference with LSTM. *Computing Research Repository, 1512.08849*, 2015.

[11] Y.Luan, Y.Ji & M.Ostendorf. LSTM based Conversation Models *Computing Research Repository, 1603.09457*, 2016.

[12] R. Rosenfeld. Two decades of statistical language modeling: Where do we go from here? *IEEE, 88(8):359–394*,2000.

[13] L.Verwimp, J.Pelemans, & P.Wambacq. (2017). Character-Word LSTM Language Models. *Computing Research Repository, 1704.02813*,2017.

A Text generation using test sample from test set [3]

A.1 RNN1: one LSTM layer, 512 hidden units

```
- Test with temperature of 1.0 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apain fair eringth.  Fur mad ip his fon look, y t well could
not bear left the house of Finn song on the Cont of the land of Erin.'

The soar waser for then came to the tor hawvor, but of the fliend would not bear
thee thought she hed yither could died.

And a fous wallit was the mearing of the walled of hes mand.'

'Bow it was the sons of Usna, eat is mained and till he were come, she was a rowald
how the word of the Wile come, for thy manther yet hur must nor with him the
Concobar.

And a fous sp

- Test with temperature of 0.5 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apain with thee to beheld that your proies be the door.

But low yo ,'arss, and lested the longer coulsed the children of Lir reached the
would song to were the hed.  So bload of the golden could daed he sawe lind, 'I sine
were in the yightsod with a gair comeand him.  Soreat she were as an this place, and
with he deart ent sorrow all that the Wine of the good, and were the children of Lir
behed him grant fight be the forest, and they were coll.

Now it was the swart comm, and if thou did he with th
```

A.2 RNN2: three LSTM layers, 512 hidden units each

```
- Test with temperature of 1.0 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apact Dermat Dercat.  And then the riven silet as bnin and to
the first, and he bsangiy, save rorring mer bod the hedned would to Bot the comd
spow the Hingy.

Bit and beinothered and fougst wot the binten as the Pare, and they dey lealp Cidms
fod the blou faar thr ereew Granis yain this hind non clat she dett suind of theme
of A dad ane I sider and Finces, And the Wil begongs was light Coum of blown, as
they sawion doread.'

Avow the als tattam to Gur a time sor, and a grieed to Nathou gamsiny at,

- Test with temperature of 0.5 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apd the wangh fow to the han bire and the pading and they sail
thou dore meaning sor might wimin Deirded the and and and beather the bud the fore
not be for the sorent by the would not beare for the would not her had did and suth
theme his hear hear and the the bocked well the fare wer the Hilled to the was the
chome have sorised for they come water sous the pain and their arle his hand to wing
wis child unto the bring houls of the her the fore.'

Then with her deed their that then blane it inter
```

A.3 RNN3: three LSTM layers, 1024 hidden units each

```
- Test with temperature of 1.0 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apl fot no maavid thmee tg reaephe by kh onhere ofAise thepeed
,hhd saoolt earialer.

so Dehtaaho e sregt n theow jDe 'nealehahd, nehtlne cl seit,f nsmid then veh
fwhinwhn, nnu ntnh wN inso oof aweaFs tia,a a riglh.  Dswe the,thdh reacht tf thret
thou s steatecost t win th s hrid honerhes overdhamh.aasred wDle onweethouegiin
aot be dv rring of nBs, atpd evousfer, spake the wird arn ished aomd, wrneYaee nsw
seadeee,Lan unoih pelaydan f thr stjaw thehafalad swaa t mn nn.n con Thite ene


- Test with temperature of 0.5 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apd thhough f y ofht maet Cir n one ere inah nc t oheoh th e
oreo hane, od e rontny wtd i nr h w unne the eol fo day nfere a c m ng herd, aeoun e
cried t ne to ure oe rth sha tdouch.'

Then Dsrt sa ded yairtae th s found w th the s f n nder sat ts mer n mee.  s ltny
f mh ran,ee ne s r th eremed f t oaten t ne r the woed,aese than is nereat,s rgo
he w t wnre bur w tahennoe fno deep ho, osee to th nos that w sheardtt tts it hist
herherd, t htroe no with Albatd
```

A.4 RNN4: three LSTM layers, 256 hidden units each

```
- Test with temperature of 1.0 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apay the Lir and with your lill upon him.

Brie the child the our erk the sorrow, and not deade this vane and land he said
Deirdre, saigh he stoud the doors.  But the sunly came words of her his befole
filled the weat lay Deirdre, said the forest and they came him them antil they were
did a truth there ye heard that the words they was not come to the quicken-tree
their they not her disted.

n the nigh morried ye shall keepech he to the stest.  Then hat sailed from Lir, and
yet us flama, be her ence

- Test with temperature of 0.5 -
That evening, too, Princess Rose went out on her balcony and clapped her hands.  But
when the golden bird apaid the his great gain, the son of Finn and him, and and it
were her be found, then left the surly, will he werl hit move to the wonder, and
there he he son.  And there the son of Finn and Dering his from the sons of Usna
came to the sons of Usna.'

And Deirdre the Green Islet thou ead in the might of yice her as the house of the
sons of Usna.'

And Deirdre the Green Isles thou of Dermat the man and its han the wing the berries
in he the the fore of to the children of Lir.  'So monger and eyes wit
```