Third Midterm

Author: Vincenzo Gargano

All code used can be found here in my fork of the torch rnn.

Brief recall on RNNs

Recurrent neural networks are best used to model sequences, some practical sequences we've seen are time-series, so signals in time, but also non-wavey sequences data like words in a book, for example Shakespeare works.

- Speech
- NLP
- Music
- Proteins

Generally when size of input is variable and size of the output is variable.

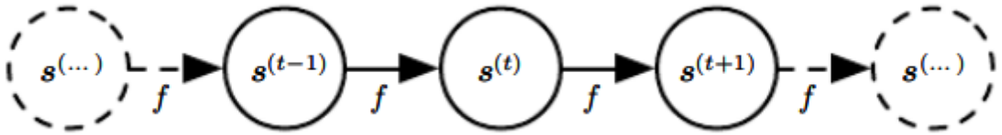
We've seen `One-to-one` basic *vanilla*, `one-to-many` vanilla unrolled in time taking an input an generating sequence by selflooping the output into itself ,and `many-to-many` (where a sequence goes in and after that ends we get an output sequence) and `many-to-many` bidirectional (sequence in sequence out for each time step).

RNNs are dynamical systems

From DL Book

$$\mathbf{s}^{(t)} = f(\mathbf{s}^{(t-1)}, \mathbf{x}^{(t)}; \theta)$$

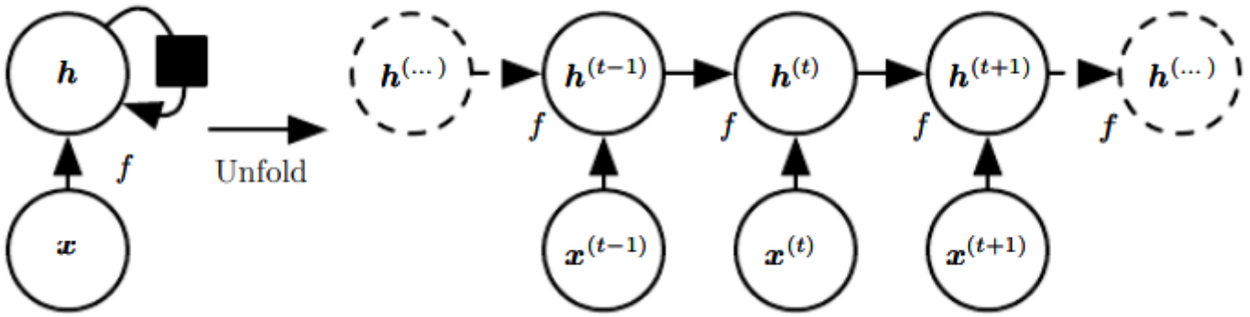
We can define our recurrent net like this, where $s^{(t)}$ is the state of our system unfolded in T time steps, and this system is driven by external signal x.



Usually hidden units in these nets are defined with this equation

$$\mathbf{h}^{(t)} = f(\mathbf{h}^{(t-1)}\mathbf{x}^{(t)}; \theta)$$

Unfolding structure will be similar to this



Backpropagation in Recurrent nets

Again basically backpropagation is the thing we use for learning from errors, changing our weights depending on the error each weights bring to the previous layer until we reach the input. In this case we're unrolling trough sequences *usually trough time*, (sometimes we can unroll in space sequences: images for example).

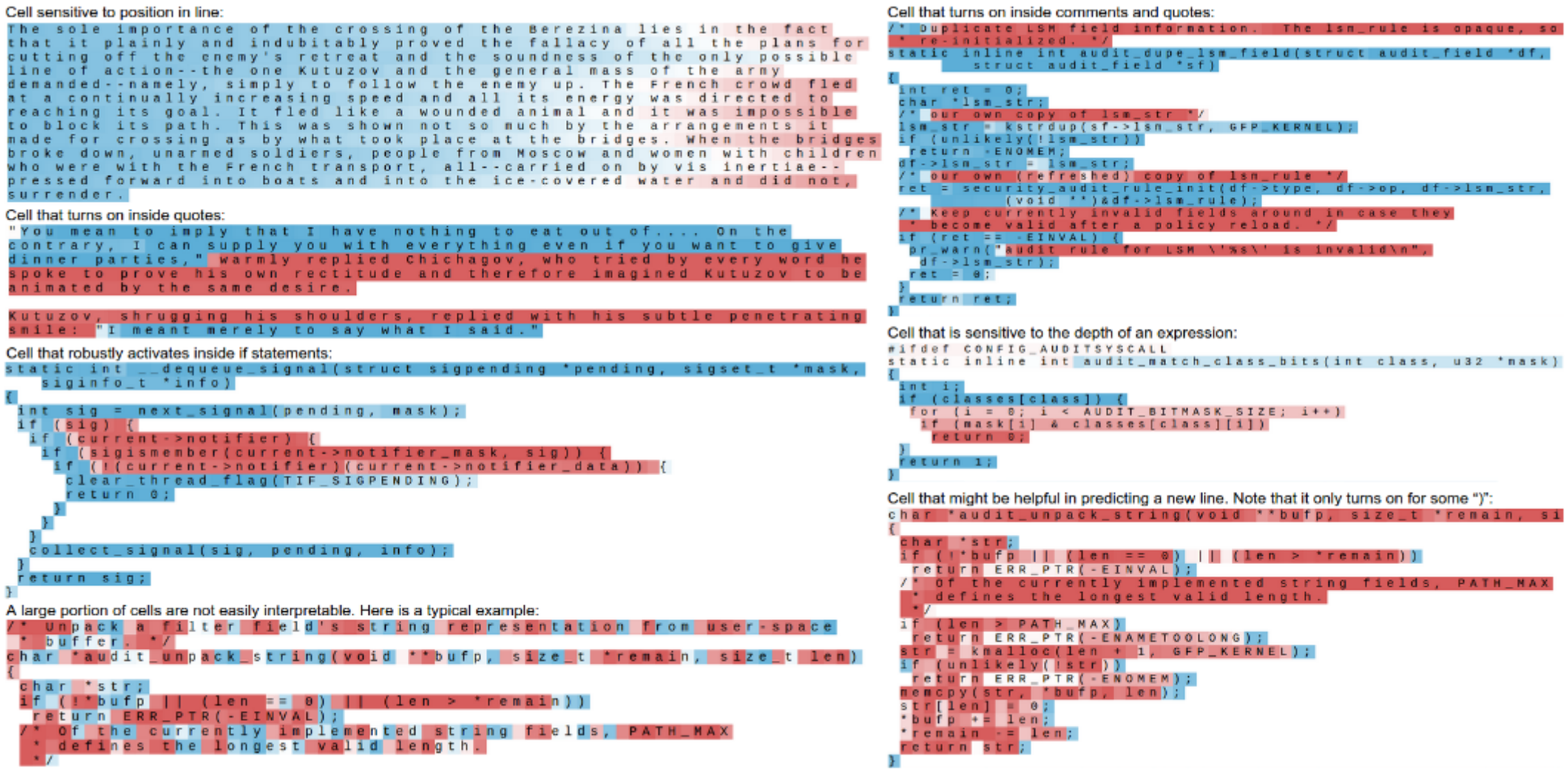
Usually in backpropagation we've ground truth the `target y`, we measure the error of our forward pass and backpropagate the gradients of this error in the net (what the error produced against what we expected).

Training is quite simple and what is optimized is a loss over all the time steps (each time step has it's loss), we sum the losses to get the total loss.

Visualization of knowledge in long-range LSTM cells.

An LSTMs can in principle use its memory cells to remember long-range information and keep track of various attributes of text it is currently processing. For instance, it is a simple exercise to write down toy cell weights that would allow the cell to keep track of whether it is inside a quoted string. However, to our knowledge, the existence of such cells has never been experimentally demonstrated on real-world data.

In the image we can see what Andre Karpathy did for visualizing what some cells are doing.



Karpathy on Parameters: Approximate number of parameters

Note

The two most important parameters that control the model are `rnn_size` and `num_layers`. I would advise that you always use `num_layers` of either 2/3. The `rnn_size` can be adjusted based on how much data you have. The two important quantities to keep track of here are:

- The number of parameters in your model. This is printed when you start training.
- The size of your dataset. 1MB file is approximately 1 million characters.

These two should be about the same order of magnitude. It's a little tricky to tell. Here are some examples:

- I have a 100MB dataset and I'm using the default parameter settings (which currently print 150K parameters). My data size is significantly larger (100 mil >> 0.15 mil), so I expect to heavily underfit. I am thinking I can comfortably afford to make `rnn_size` larger.
- I have a 10MB dataset and running a 10 million parameter model. I'm slightly nervous and I'm carefully monitoring my validation loss. If it's larger than my training loss then I may want to try to increase dropout a bit and see if that heps the validation loss.

Or Lercio Headlines dataset is 475kb to choose the best rnn_size and num_layers

Temperature and Perplexity

Temperature is an hypeparameter used to generate new samples, for higher temperature values the net tend to be more "free", so manage to write a lot of different things, instead for low temperature the generated samples are more likely to repeat because mathematically what we're doing is the following thing:

Since neural network produce probabilities using softmax output layer converting *logits*, z_i (aka unscaled output of previous layer) into probabilities q_i.

Given a temperature T

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T)}$$

Higher the value of T produces soft probabilities distribution over classes so basically we are doing this $\frac{logits}{T}$, comparing different logits to get a probability distribution

In the code we have this

```
# Sample from the network as a multinomial distribution
output_dist = output.data.view(-1).div(temperature).exp()
top_i = torch.multinomial(output_dist, 1)[0]
```

So getting output_dist and then picking the most probable character from it.

More abstractly we can summarize the Temperature in general for Language Models as the confidence of our model in generating the next sample, higher value of temperature make the model more sensitive to small values this gives more diversity to our generated samples but usually we pay with more errors, like words that do not exists or loosing context information. Lowering temperature results in the probability of some samples becoming higher, so the probabilities of sequence piece tends to 1, resulting in much less diversity in our sampling.

Perplexity instead is a measure we encountered in the HLT course Lesson 2 - Language Modeling > Perplexity, and basically tells us how suprised is the model on the new data, we would like to minimize the perplexity.

Mathematically we can define perplexity measure like this

$$PP(W) = 2^{H(W)}$$

Where $H(W)$ in information theory is the Entropy of our RV defined as $\frac{1}{N} log_2 P(w_1, w_2, \ldots, w_N)$

Char-RNN

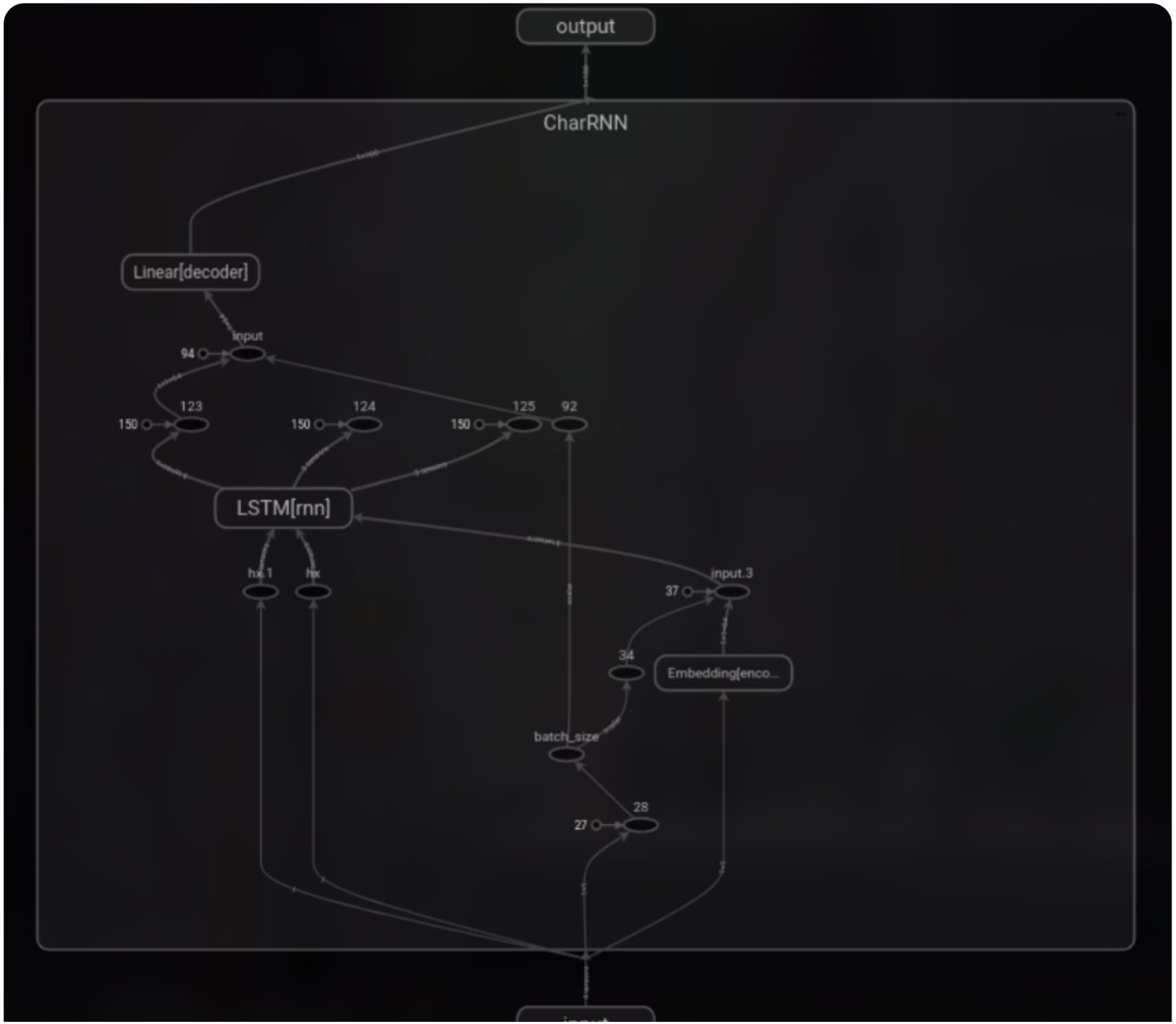
What are we trying to do is to learn a probability distribution over our data on "what character should come next?".

Experiments

For the learning rate i used all the times the same classical 0.01, and Pytorch uses some optimization algorithm like Adam to get an adaptive learning rate.

Let's visualize results of some experiment i tracked using Tensorboard

This was the "generic graph structure" used in almost all experiments


output
CharRNN
Linear[decoder]
input
94
123
150
124
150
125
92
150
LSTM[rnn]
hx.1
hx
input.3
37
34
Embedding[enco...
batch_size
28
27

Experiment 1: Lercio Headlines

I trained two different models one was an LSTM with 100 hidden units and circa 181k parameters

Results: Char RNN over Lercio Headlines

During training temperature was setted pretty high: 0.8

What i was during training was that initially, after first epochs, the output is pretty poor, the model doesn't undestand where to place letter and in what order, after a while the model is capable to put the quotes at the right spot, so at the start and at the end of a phrase or a sentence. Or maybe sometime when citing someone speaking. Using the constructs `Name: "citation"` or `Subject:"news about it"`

```
Training for 2000 epochs...
  5%|▌                                                                                | 99/2000 [00:45<14:44,  2.15it/s]
[0m 46s (100 5%) 2.1773]
Who di mortata per all'Oal apre dattino di par due lei e non scomalla divessaste con non con praccore

 10%|▉                                                                               | 199/2000 [01:35<14:29,  2.07it/s]
[1m 36s (200 10%) 1.9730]
Who di piulistimo scopera di fiscrere del'ore"
!scoprieri la sessori di gazzi di si giorgiaghio: "Opo

 15%|▍                                                                              | 299/2000 [02:26<15:42,  1.80it/s]
[2m 26s (300 15%) 1.8713]
Wha la rimente del carta dei sue solari con alla stragoli dal Svista la castatera
"Casoltico pardubano

 20%|▊                                                                              | 399/2000 [03:19<14:19,  1.86it/s]
[3m 20s (400 20%) 1.7989]
Wham
"Sta Malfato quelli in accuola di crenita di mende. Gli dovro vicina e il potrendiaro
La Crininit

 25%|▎                                                                              | 499/2000 [04:13<14:01,  1.78it/s]
[4m 13s (500 25%) 1.7889]
Whi con la due chiabio
Facebook senzina sul conter Puta non si butta dei gli e oltro recimi
"C'elegola
```

LSTM:

A more refined model output changing the temperature (lowering it).

```
master ? !
  λ  python3 generate.py models/model_lstm_2_layers.pt --t 0.4 --prime_str "Calcio: Napoli vince lo scud" --predict_len 500                  char-rnn.pytorch
Calcio: Napoli vince lo scudetto della serie di Geova alla fine del cancro"
"Torino con un altro per il primo la piloti della Cristo del Papa Francesco in cui i suoi giorni di Papa Francesco e parte in mette
"Stamina alla cartella il cancro che la scelto di Papa Francesco e perde della prossimo non riesce il primo contro l'alberi di vince alla fine di appare il nuov
o pagato della solitario e si scopre la presto e lo per farci"
"Compra il posto di figlia il primo come scopre che la sua campo di contermerano il primo come di                                          /2.3s
```

Again let's try with high temperature at the end of training : temperature set of 0.7

```
  λ  python3 generate.py models/model_lstm_2_layers.pt  -t 0.7 --prime_str "TGLercio: " --predict_len 350
TGLercio: "Bospera il figlio con un caffe gli la colore del giornato di Siciliaio"
JOTTIVITA PER A FAVER SCOPERTO"
Padre Piero Arone. Tirone salta di celleranze la strangiato ma per i pittili
Arrestato da loro getta in Greppia"
Soologia: "Ti forma per devi dannalesi con un posta a disociazione
"Sanremo. Salvini la stermine di carco di Battisti: specchio tutt
```

As we can see the model is making up new words, this is because of the char-by-char generation, temperature make it more creative but sometimes we want that tradeoff between creativity and fidelity to data. From what i sperimented the ideal temperature reside in something from 0.4 to 0.5.

GRU:

Low-medium temperature:

```
master ? !
  λ  python3 generate.py models/model_gru_2_layers_256h.pt --t 0.5 --prime_str "Scuole chiuse a napoli:" --predict_len 500                  char-rnn.pytorch
Scuole chiuse a napoli: "Non si sono i cantieri per farli intervistato di "Chi sostituiscono da un foto dell'associato di miglior in citta si asposta di Geova
Si al come in cui con la mamma per assume con la moglie di marica di campione di mangia in campiona
"Nasco compara a Basta Mondia, ma solo in campionato in cocanta con la commercia da una cramero della sosticolo dalla lingua al contacca di Gianni"
"Esplota di Mara Cancisto di Giovane si tocchiare il commentare""
"Giovane in complotto di Ginerminato in campione a                                                                                          /2.9s
```

Higher temperature:

```
 λ  python3 generate.py models/model_gru_2_layers_256h.pt -t 0.7 --prime_str "TGLercio: " --predict_len 350
TGLercio: Gesu compra scara batteria
Bol suo cambia come stato da un romanzi di riesciato del suo accordo
Dal suo commercia in mamme che incontatti inventa si si uccide il suo salo italiana che si uti da un cano i bambini mondiali di Finanzia
"Roma, scienza letto di signo del PD"""
""Dopo nuovo piazza la scompora espirato un moto""
Roma: "Non esista a Rapina
```

Reading these generation i expeted better results from

Experiment 2: Drosophila Melanogaster Genome

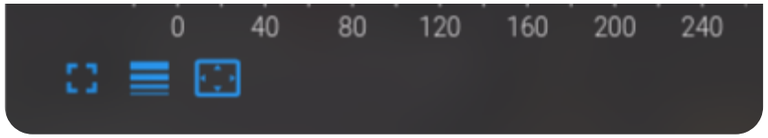
Model used here was an LSTM with 64 hidden units over 250 epochs, for a total of 79460 parameters.

Loss

Loss/train
tag: Loss/train

Perplexity:

Perplexity:
tag: Perplexity:



Results : Char RNN generating drosophila genome and aliging with BLAST

Result I got by sampling from the CharRNN, with low temperature: What i done is generating some sequences then feed them to the Protein\Genome similarity search website and wait for results. The metric used by this website to quantify how much similar they are is the E-value, which evaluation depend from the dataset and the lenght of the data user gives. Smaller the value better the similarity, what i think is happened here is that i picked a small piece (500 character sequence) and fed it to the database, what i got is a very similar piece to the `chromosome 3L` of the Drosophila, i tried with other high temperature and i got a similarity with some part of the entire chromosome X, but the E-value in that case was pretty high: 0.041.

Align.	DB:ID	Source	Length	Score (Bits)	Identities %	Positives %	E()
1	EM_INV:AE014296	STD:Drosophila melanogaster chromosome 3L. *Cross-references and related information in:* ▶Genomes & metagenomes ▶Literature ▶Protein families ▶Nucleotide sequences ▶Samples & ontologies ▶Protein sequences	28110227	54.1	62.0	62.0	7.3E-6
2	EM_INV:AE014296	STD:Drosophila melanogaster chromosome 3L. *Cross-references and related information in:* ▶Genomes & metagenomes ▶Literature ▶Protein families ▶Nucleotide sequences ▶Samples & ontologies ▶Protein sequences	28110227	39.7	79.1	79.1	0.16
3	EM_INV:DS485250	CON:Drosophila melanogaster chrUn.1708 genomic scaffold, whole genome shotgun sequence.	1192	38.9	54.8	54.8	0.28
4	EM_INV:DS483844	STD:Drosophila melanogaster 211000022280457 sequence.	4471	38.9	54.8	54.8	0.28
5	EM_INV:AE014297	STD:Drosophila melanogaster chromosome 3R, complete sequence. *Cross-references and related information in:* ▶Literature ▶Protein families ▶Nucleotide sequences ▶Samples & ontologies ▶Protein sequences	27905053	38.6	71.9	71.9	0.34
6	EM_INV:AE014298	STD:Drosophila melanogaster chromosome X. *Cross-references and related information in:* ▶Literature ▶Protein families ▶Nucleotide sequences ▶Samples & ontologies ▶Protein sequences	23542271	38.6	64.8	64.8	0.34
7	EM_INV:AE014298	STD:Drosophila melanogaster chromosome X. *Cross-references and related information in:* ▶Literature ▶Protein families ▶Nucleotide sequences	23542271	37.5	55.5	55.5	0.73

		▶ Samples & ontologies ▶ Protein sequences					
☑ 8	EM_INV:AE014297	STD:Drosophila melanogaster chromosome 3R, complete sequence. *Cross-references and related information in:* ▶ Literature ▶ Protein families ▶ Nucleotide sequences ▶ Samples & ontologies ▶ Protein sequences	27905053	37.5	65.8	65.8	0.73

Experiment 3: Music generation

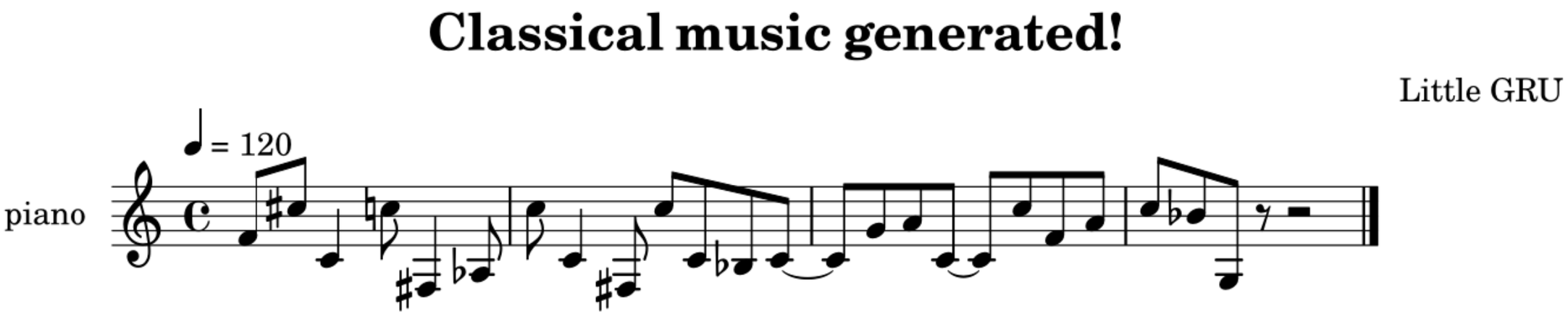
Here i searched for the most simple and big dataset i could get my hands on, i tought to convert a large portion of midi to text, but didn't knowing really how to do this in the current time i just took this dataset

This time i tried both a LSTM and a GRU, and i obtained nice results with the GRU!

Results:

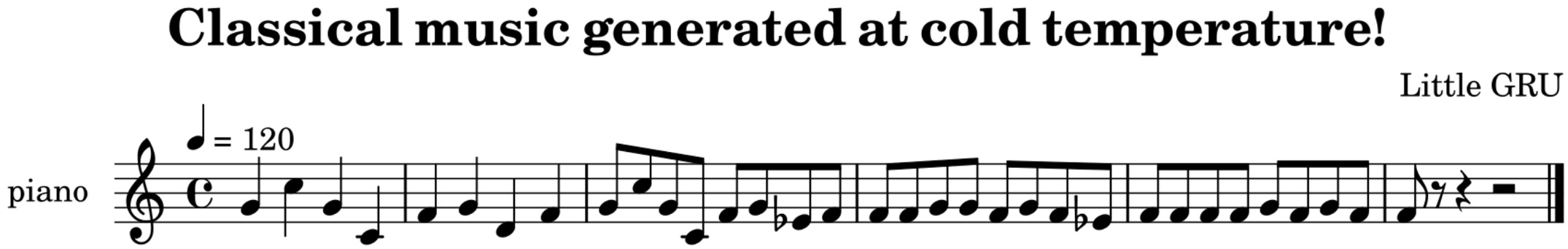
I just asked the net to generate the sequence of notes then i wrote a little script using a library called SCAMP to generate the scores. I just added dynamic like loudness and duration.

This sample was generated with 0.5 temperature and the start is quite messy, but then further it became pretty nice the melody towards the end.

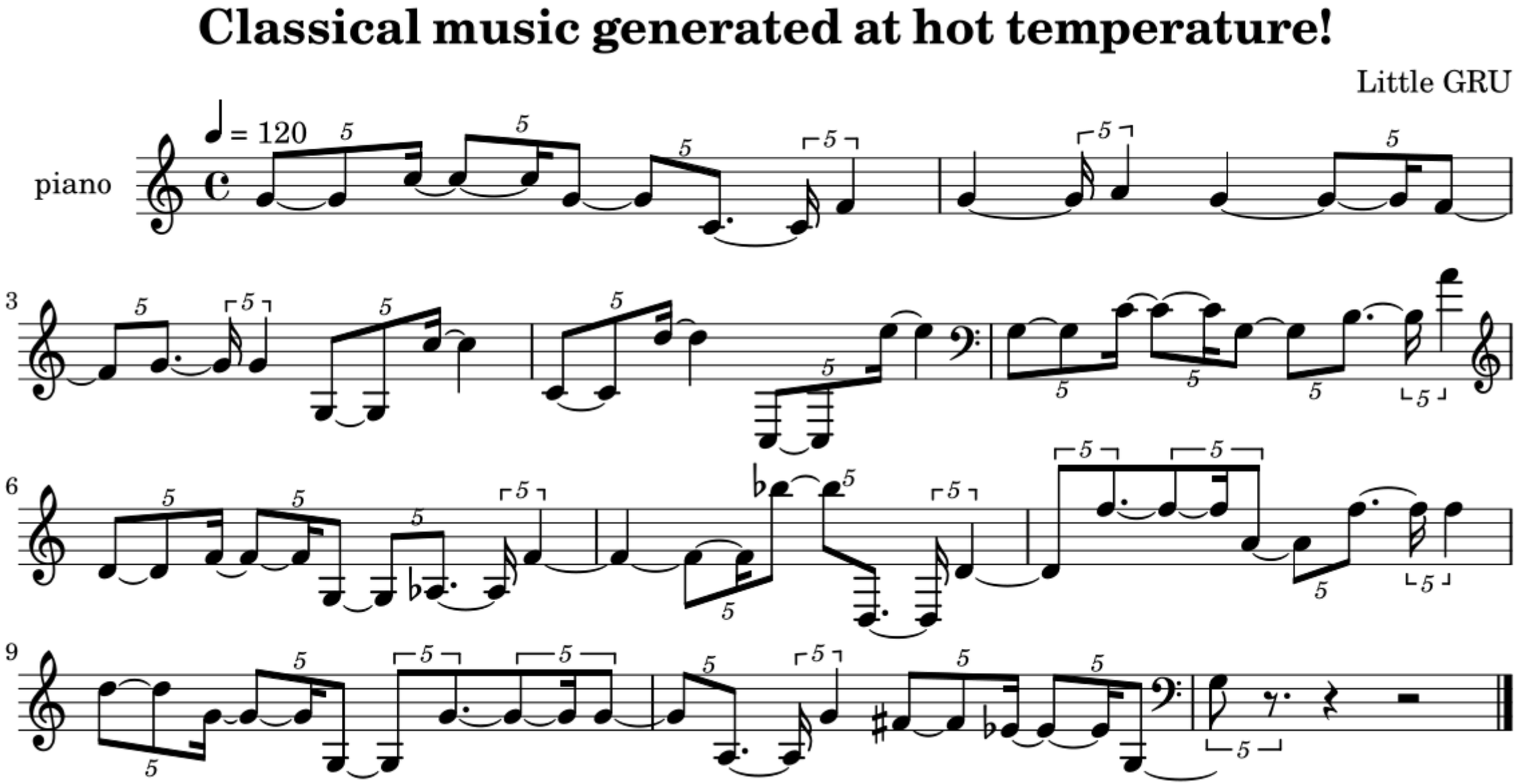


Unfortunately on pdf i can't insert sounds here.

Now let's try with a melody i use as prime string to the net and just lowering the temperature



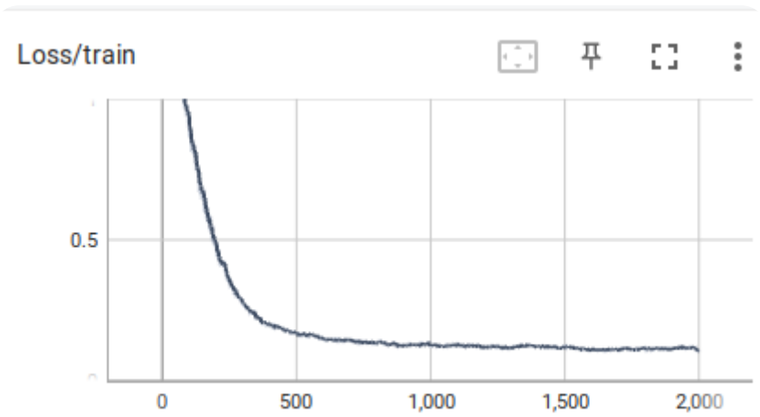
And what if we crank up the temperature?

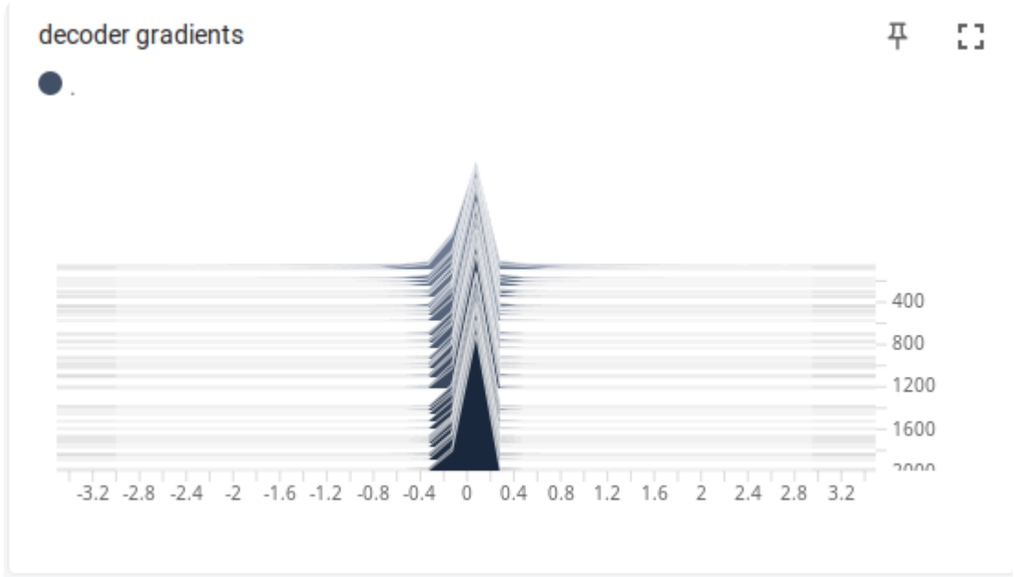


Yeah the piece becomes more jazz than classical music. But there are certain parts where the GRU come up with some octaves that's really nice, its in the 3 measure.

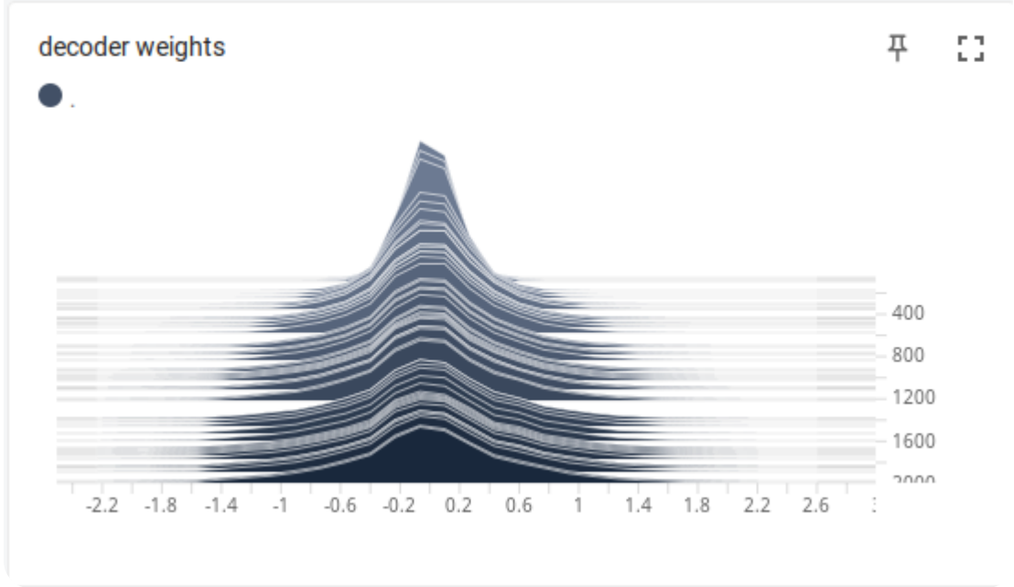
Interpreting results with Tensorboard

Some plots of our loss, gradient and weights histograms from Lercio training.

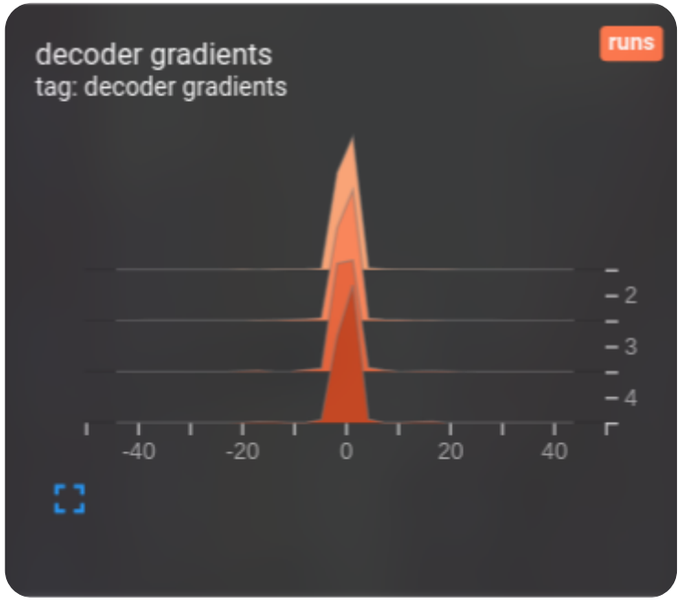




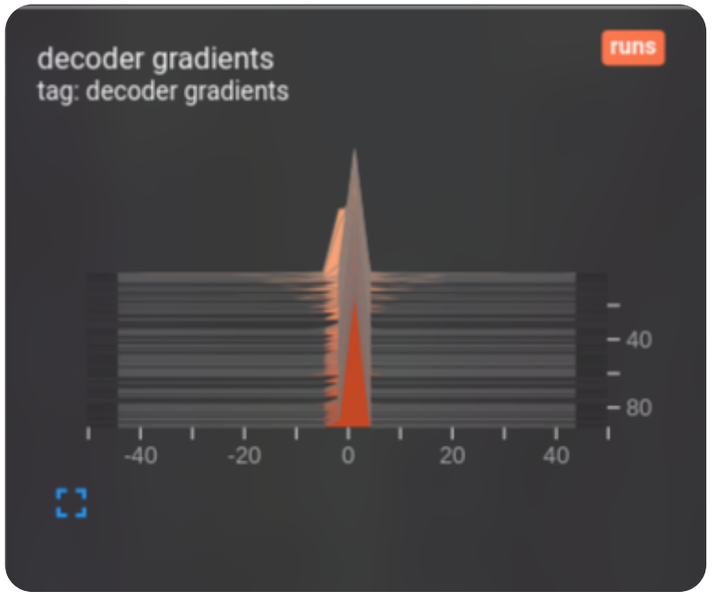
decoder weights

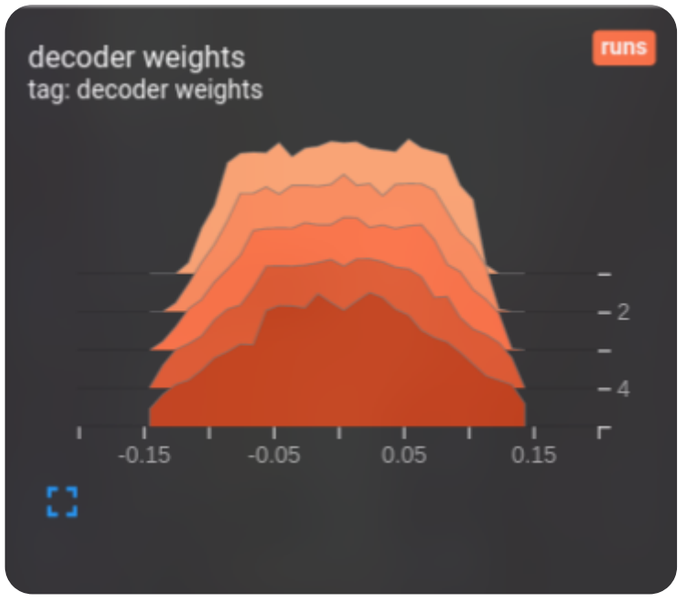


Let's see the gradients and weights at the **start** and **after a little** while of training of the train on another model and different data.

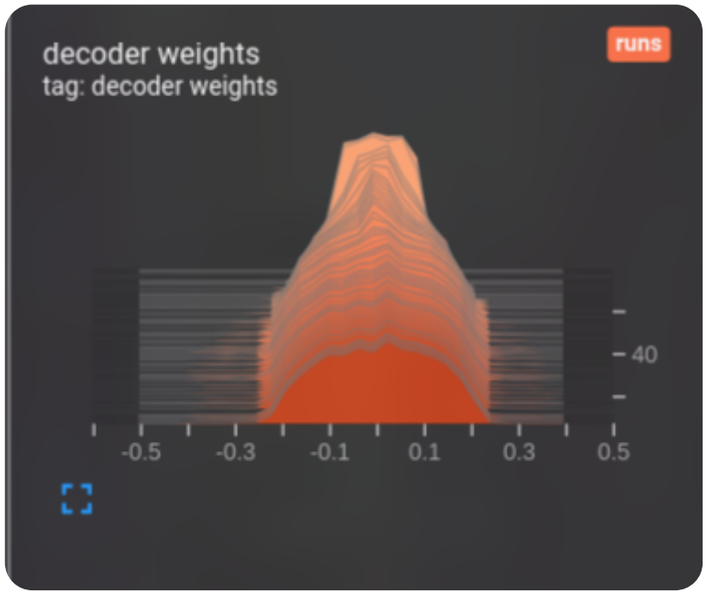


After some epochs...


decoder gradients
tag: decoder gradients
runs
40
80
-40
-20
0
20
40


decoder weights
tag: decoder weights
runs
2
4
-0.15
-0.05
0.05
0.15

After some epochs...



So by exploring these plots i understood that initially the distribution are more irregular but still near a normal distribution, probably because of initialization, then these tends to become rounded or generally tuned toward the training process, since in the RNN weights are shared trought time steps.

For the gradients we we oberve a spike, these can we observe that are greater at the start and becomes smaller **"more spiked"** so narrowed their values because of the optimization making progress for the learning.

Conclusion and future works

RNNs, but more intrestingly LSTMs and GRUs are capable to work on sequences, and we have a lot *different* data sequences that can feed them, characters in language, musical notes, DNA and Protein sequences and many more. What we can expect is to draw new but, a lot of time nonsensical samples, from a distribution that is mimiking our original from what the data we feed them.

What if we want to make generation more affine to our language? Well we need to give some **context** information to our network, so that it understands the connections between the words, and this is what the **Attention** mechanism do! Basically what attention do is to improve the encoder-decoder by allowing decoder to utilize the relevants part of the input sequence by *weighting* the encoding input informations.

And finally we can talk about the limitation of these RNNs, widely known is the slow and unparallelizable training of these sequential model that gave rise to the Transformers architecture. More is that very long term dependancies are lost and we can encounter problems like **vanishing gradients**.

But still these models are powerful for modelling sequences especially if time is involved in fact transformers uses *positional encoding* to deal with the temporal order in input sequences. Also RNNs have some internal memory and can work with information more "naturally" than the new architectures, they are more simple and more similar to how biological brain works with retroaction of the signal that comes back as input.

Resources

The Unreasonable Effectiveness of Recurrent Neural Networks blogpost from which Andrew Karpathy explain and reason about how recurrent networks...

From Visualizing And Understanding Recurrent Networks by Karpathy Temperature explained by Hinton

Drosophila experiment website for segment similarity

Faq

Protein Similarity Genome similarity