Alessandro De Grandi

Deep Learning Lab - Assignment 3

1. Language Modeling with RNNs:

1.1 Properties of the data:

1.1.1

Vocabulary size = 107 (keys from 0 - 106 , 0 is used for padding, 1 for unknown characters).
Total number of characters = 177517
Number of lines = 5033
Chars per line average ≈ 35 (short lines, a lot of empty lines)
There are a lot of short paragraphs.

1.1.2

Reduce the vocabulary size while maintaining the same meaning:
Contractions could be expanded.
Lowercasing.
Maybe remove very low frequency characters (eg: proverb—μαλλον ὁ Φρὺξ , useful?)

1.2 Batch construction

1.2.1

The function get_idx is used to get the id of a token from the vocabulary. If the token is not yet in the vocabulary, it will extend it, adding the new token (call to add_new_word). Otherwise if the option to extend the vocabulary is false, it returns the id used for the unknown token.

1.2.2

In id_to_string: the key is an integer (id) mapped to a character value.
In string_to_id: the key is a character mapped to its id value.

1.2.3

The number of characters in the input file (177517).

1.2.4

The number of batches(87).

1.2.5

From TORCH.TENSOR.NEW_FULL documentation:
`Tensor.new_full`(*size*, *fill_value*, *dtype=None*, *device=None*, *requires_grad=False*)
Returns a Tensor of size `size` filled with `fill_value`. By default, the returned Tensor has the same `torch.dtype` and `torch.device` as this tensor.

First line:
Returns a tensor of size segment_len*batch_size filled with pad_id (torch.Size([177536]) all 0).

Second line:
Input_data is a LongTextData object, .data contains all of the token ids in ‘book order’.
Fills the padded tensor with the data (id of tokens).
Padded looks the same as .data but with 177536-177517= 19 zeros.

1.2.6 padded[0:bptt_len] is shape padded[0:64] that is [64] (batch will have 0 at start)
1.2.7 padded[i * bptt len - 1:(i + 1) * bptt len], last element of previous batch is included so shape is [65]

1.3 Model and Training

1.3.1 Implemented the model as a class with constructor parameters:
Number of embeddings, hidden state size, number of Lstm layers, vocabulary size.
Eg: (64, 2048, 1,107)

```
LSTMModel(
  (embed): Embedding(107, 64, padding_idx=0)
  (lstm): LSTM(64, 2048)
  (linear): Linear(in_features=2048, out_features=107, bias=True)
)
```

1.3.2 Implemented a decoding function, which makes the model “read” the prompt first, updating its internal states. Then uses the last character of the prompt to start predicting the new sequence and feeds each character output back into the model.

1.3.3 The decoding function also has an optional parameter to allow sampling during decoding, it is set to default true; if set to false the function will use argmax.

1.3.4 Implemented training, detaching hidden states tensors before calculating the gradients. Monitoring every 20 steps, printing the perplexity and decoding the prompt:
“This assignment is” and producing 10 characters. (with sampling enabled)

1.3.5 Trained the model with the given parameters, stopping when the perplexity reaches a value <1.03.
The model starts with high perplexity and it is producing nonsense in the form of random characters:

```
Epoch [1/100], Step[1/87], Loss: 4.6730, Perplexity: 107.02
This assignment iso0—hhosId:
```

The perplexity quickly decreases and the model produces nonsense in the form of words:

```
Epoch [2/100], Step[61/87], Loss: 1.8865, Perplexity:  6.60
This assignment is gigle viv
```

At some point the model starts to produce meaningful words:

```
Epoch [10/100], Step[1/87], Loss: 0.7274, Perplexity:  2.07
This assignment is not a Fox
Epoch [10/100], Step[81/87], Loss: 0.6356, Perplexity:  1.89
This assignment is accepted
```

End of the last epoch:

```
Epoch [30/100], Step[61/87], Loss: 0.0292, Perplexity:  1.03
This assignment is strength.
Epoch [30/100], Step[81/87], Loss: 0.0319, Perplexity:  1.03
This assignment is a present
Perplexity<1.03 stopping..
```

The model consistently generates meaningful words; sentences are still of dubious meaning.

1.3.6

TEST 1: Increased learning_rate = 0.002
The perplexity is oscillating, it takes many more epochs for it to converge to a value <1.03. (51 epochs)

TEST 2: Decreased learning_rate = 0.0005
The perplexity steadily decreases and it takes fewer epochs to converge to a value <1.03. (25 epochs)

(back to learning_rate = 0.001)
TEST 3: Decreased bptt_span= 32 number of batches is now 174
Not noticing much difference in output quality.

TEST 4: Increased bptt_span= 128 number of batches is now 44
Not noticing much difference in output quality. It might be better for completing longer sequences.

1.3.7 and 1.3.8

a)Prompt = “THE HARE AND THE TORTOISE” , output length = 500 :

ArgMax:

```
THE HARE AND THE TORTOISE


A HARE one day made himself merry over the slow passed on the house itself
but because she had not contrived iron wheels in
its foundation, so that its inhabitants might more easily remove if a
neighbor proved unpleasant.

Indignant at such inveterate fault-finding, Jupiter drove Momus from
his office as judge and expelled him from the mansions of Olympus.



THE FATHER AND THE SNAKE


ONE wintry day a Farmer found a Snake lying on the frozen ground, quite
stiff and nearly dead with cold.

I
```

Sampling:

```
THE HARE AND THE TORTOISE


A HARE one day made himself merry over the slow of the Trees, set to work
first to draw a heavy
clog, which stuck in his throat and presertt of foolightingale tages
not protected by copyright lake. At last caught Pitcher and
hardction or the help of the courses to be content here. There is water
part

“Curt of the prover, and now to death. Unew you have finished down upon
the cause of the
action. So he called out as loudly as he could, “Help! help! The Wolf!”
and all
the Men came running us.”
```

b)Prompt = “THE STUDENT AND THE PROFESSOR”, output length = 500 :

ArgMax:

```
THE STUDENT AND THE PROFESSOR


A CROW stole a piece of cheese out of a fire that was
burning in the woods, where the trees hung over the
water, and where no one came to disturb him.

The other lived in a small pool. This was not a good place for a Frog,
or any one else, to live in, for the country road passed through the
pool, and all the horses and wagons had to go that way, so that it was
not quiet like the pond, and the horses made the water muddy and foul.

One day the Frog from the pond said to the other, “Do come and
```

Sampling:

```
THE STUDENT AND THE PROFESSOR


* YOU DEStice, Donkey in a bart for its big Wompalakes, in the
distribution of
electronic work, or any part of this electronic work, without
prominently displacing them with the gree you distort, perhers for
free
  dispation, how to help them, “and have lived long
enough. You disturb every one in the house by your loud crowing in
the
trademark of the Ants asked him how he had disposed of his time in
the
world.”

“And so would we,” said the cool. “I should like to see you
dance before I die.”
```

c)prompt = “TWO men were traveling to”, output length=250

ArgMax:

```
TWO men were traveling together.

“Now,” said the Father, “if you wish for from never seen but a
poifter said to a Baby.”



THE CAT AND THE MARTINS


A CAT, hearing that some Birds who lived in a martin box near by were
ill, put on his spectacles and his overcoat, and made
```

Sampling:

```
TWO men were traveling together.

“No,” replied the Goat; “you forgot to help me out.”

“You found that it was flax as the meadow. If she is better
than often forgerments of the house, and jumped up on his
master’s king charge-1 in which they are delicious. You are some one
```

d)

prompt=”Anything you think might be inter”, output length=100:

Argmax:

```
Anything you think might be interfort. Why, you will have
no part with a Man who can blow hot and deep that, though he had little
cha
```

Sampling

```
Anything you think might be interpolated us from
his office as judge that no material was so good for the
purpose as began to reach it. “Many people well to laugh;
```

The output can be novel, but not very meaningful. The model is producing meaningful words most of the time but the sentences often don't make any sense, and when a sentence makes sense it is probably by chance or because they are the same as in the book, especially using agmax, the model outputs memorized sequences.
So the model does not produce fables, but it could be useful for word auto completion.

1.3.9 BONUS:
Copy pasted a bunch of articles from the internet into a text file:

- https://towardsdatascience.com/text-generation-with-bi-lstm-in-pytorch-5fda6e7cc22c
- https://medium.com/coinmonks/character-to-character-rnn-with-pytorchs-lstmcell-cd923a6d0e2
- https://blog.floydhub.com/long-short-term-memory-from-zero-to-hero-with-pytorch/
- https://pytorch.org/tutorials/beginner/pytorch_with_examples.html
- https://medium.com/analytics-vidhya/no-need-to-be-perplexed-by-perplexity-cd4cb71ac97b
- https://en.wikipedia.org/wiki/Perplexity
- -https://machinelearningmastery.com/sequence-prediction/
- https://machinelearningmastery.com/exploding-gradients-in-neural-networks/
- https://en.wikipedia.org/wiki/Natural_language_processing
- https://analyticsindiamag.com/complete-guide-to-exploding-gradient-problem/
- https://www.topbots.com/perplexity-and-entropy-in-nlp/
- https://medium.datadriveninvestor.com/how-do-lstm-networks-solve-the-problem-of-vanishing-gradients-a6784971a577

Trained the model with this new dataset and asked it to finish this assignment for me:
(Generating multiple answers with both argmax and sampling and picking the one I liked most)

2 Questions:

2.1
Prompt = "`What is the perplexity of a language model that always predicts each character with equal probability of 1/V where V is the vocabulary size?`"
Model answer (max length = 10, using argmax): `267,575 b`

My answer: V

2.2

Prompt = “In this assignment, we looked at the next-word prediction in text as a sequence prediction problem. Give two other examples of sequence prediction problems that are not based on text. Examp“

Model answer(max length = 150, using sampling):

```
le of a Sequence Pradiction Problem
Example of a Sequence Prediction Problem.png
Example of a Sequence-to-Sequence Prediction Problem.png
Example of a Sequence Prediction Problem
```

My Answer:

Time series prediction, eg: stock market.

Sound sequences prediction, speech recognition.

2.3

Prompt = “What is the vanishing/exploding gradient problem? The vanishing/exploding gradie”

Model answer(max length = 500, using argmax):

```
nts. This problem arises when back-propagating through the RNN during
training, especially for networks with deeper layers. The gradients have
to go through continuous matrix multiplications during the
back-propagation process due to the chain rule, causing the gradient to
either shrink exponentially (vanish) or blow up exponentially (explode).
Having a gradient that is too small prevents the weights from updating and
learning, whereas extremely large gradients cause the model to be
unstable.
```