

Foundation Models CS 839

Assignment 1

September 29, 2024

Sujay Chandra Shekara Sharma
schandrashe5@wisc.edu
Wisc ID: 9085871714

NanoGPT Experiments

1. Setup and Reproduction

Forstameds feet their Reep our mein him.
HUMINIO: He that the she good my the doth?
LONTES: Hake which it impery way ar he would, a sights That a didht thy will,
well thou is of sulbust.

2. Hyperparameter Experimentation

The lowest validation loss that I was able to obtain was 1.5070. The hyperparameter configuration that obtained this loss is detailed below:

Block Size: 128

Batch Size: 32

Number of Layers: 4

Number of Heads: 4

Embedding Size: 256

Dropout: 0.0

Time Taken: 221.26 seconds

step 2000: train loss 1.2478, val loss 1.5070

Similar validation losses were also obtained for other configurations with a block size of 128 and batch size of 32. These two hyperparameters seemed more impactful on the loss than number of heads/layers.

3. Evaluation Metrics

- (a) I decided to use Perplexity as the general evaluation metric as it is much more interpretable than the cross-entropy loss. It gives us an idea of how confident our model is at predicting the next character/word token. It gives us an idea of how many tokens our model is considering on average each time it generates a new token.

Computing the perplexity of the model for the hyperparameter configuration obtained in the previous question results in a perplexity score of 4.5133. This can be calculated by taking $e^{1.5070}$ which was the validation loss obtained earlier.

step 2000: train loss 1.2478, val loss 1.5070

step 2000: train perplexity 3.4826, val perplexity 4.5133

- (b) I decided to use Jensen-Shannon Divergence (JSD) as the specific evaluation metric as it gives us an indication of the similarity between the generated data distribution and the training distribution. It also produces a symmetric bounded metric between 0 and 1 which is more interpretable.

Computing the JSD score of the model for the hyperparameter configuration obtained in the previous question results in a JSD score of 0.3034. This indicates that the two distributions are somewhat similar but there are still significant differences between them leading to some room for improvement with a better model/training.

step 2000: train JSD 0.2720, val JSD 0.3034

4. Dataset

The text dataset that I obtained is from Robert Jordan's Wheel of Time series. After preprocessing this dataset, I passed a variable number of input characters to the model and then trained the model on this data. The number of input characters were varied from 250 to 512000. These are the plots that I generated:

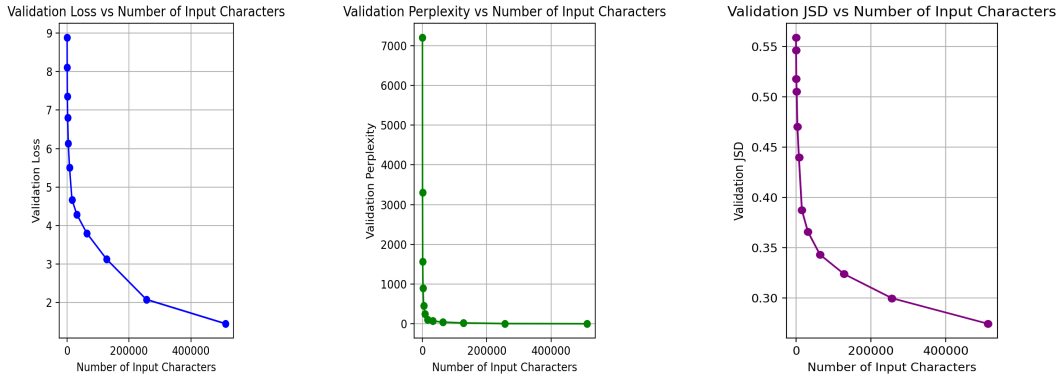


Figure 1: Plots of validation Loss, Perplexity, and JSD vs the number of characters

As observed from the first plot, as the number of characters in the training dataset increase, the validation loss decreases significantly. This suggests that as more data is provided, the model improves its generalization to unseen data. The curve starts to flatten out around 512,000 characters which is half the number of characters in the Shakespeare dataset, indicating slightly diminishing returns from adding more data past this point.

From the second plot, we can see that the perplexity drops dramatically with an increase in input characters and stabilizes around 128,000 characters. The steep initial drop suggests that early data is crucial for improving the model, while additional data leads to only marginal improvements in the certainty of the model's prediction.

The last plots shows that the JSD shows a more steady decrease as more input characters are used in training, reaching a stable value of 0.30 around 128,000 characters. This indicates that the model's predictions are becoming more aligned with the true data distribution as the training dataset grows.

Based on the metrics defined in the previous questions, we see a reasonable score somewhere around the 100,000 character mark or 1/10th the size of the Shakespeare dataset. However, we first start to see generated samples that are reminiscent of Wheel of Time at 64,000 characters. The generated samples contain the names of characters and places from the novel at this stage. Below are some of the generated samples corresponding to different number of training characters:

4000 training chars: [loss 6.1348, perplexity 461.6516, JSD 0.4701]

wolaut hind he ead his beay now the sto han shear, oughed flan the avis tought
He sun fanLis, oorince tore a bliling.

Withat ha oulout ouchilernd aeand bear, thouNo he olannd ewor. He tlout he ma
fla grieound thend luche good this feeld noud by comparill'so -comoaks laand,
ig boneded wind to the mut riagonce, the a ond a wierny to bearn to geen to
go feen the weer the sorn ithe, bun its if that. Scane t lanerled whit bours from
umpleest if walvely antong the the suof Tam wivelves to the bee

64000 training chars: [loss 3.8013, perplexity 44.7586, JSD 0.3431]

“Wit, any it? Emond’s Field. Not evincing Cight self to look how do this for
good ten hood the was a cart, noth is if it.”

“I was I sur, Mat’s not know.” Rand, but rush appoce swe sunly old the fat by
realer tught unman roWe to exace. “What?” Them ouse good on the Two Rivers,
if your han his ince. There’s good ater all over to be lush wall. She’s stle tang.”

“Well,” Mat man saide, mut off. Nobody only. Only you tell than Tam was a
tere. But is out was the Lady Moneybe my nomen to Tam Mat ab d

1.2M training chars (Same size as Shakespeare): [loss 1.2747, perplexity 3.5778,
JSD 0.2654]

“None of Bel Tine. Rand’s walking under his boots, and anger she was just that
evening the quiet. And she rocked out the horse where the sap the same in the
onlot.

Captain was still are too dream and down a stone face behind the Wisdom held
Mat her. “What is rap?”

“As was well only a bed way out of the torn of horses, a rubbed gold in you. If
this is saying on the cold, and nerry around the creak of the Palace, softly did
not even find the north in the fire. And something in the door arm in t

5. Fine-tuning

In order to go from Shakespearean output to output resembling something from
the Wheel of Time, I needed to fine-tune the trained Shakespeare model for 2500

iterations of training data using the default hyperparameters. Each iteration of training was performed on a batch size of 32 and a block size of 128.

One surprising thing is that the generated samples first go from something Shakespearean to total gibberish after a small amount of training iterations(100). However, at this point, the structure of the generated samples is similar to that of a play. After further training, the generated samples slowly turn to prose and then they become similar to the Wheel of Time. Below are some examples of the generated samples at various stages of fine-tuning.

Sample after 100 fine-tuning iterations:

RfrjB bmfy xjw—nhj fx jqxjD

OMRK VMGLEVH MMBB S: yjqj rj: M mf—j ltjx r“ ymwtzlmyD

PSVH JMYebEYIVB bjqq: nrutwyjsz gzy st rtwj: j—js ns xmj

Sample after 1000 fine-tuning iterations:

bjqq: fx M xf“ f wjxy ns xtqinju: fsi fx rtwj rfwnfljx: fsi xjw—nsl ymwjj ‘nym rj: fsi ymj xznytwx tk mnx xmtzqi pjju ymj tzyrnsfyj: fsi xmj tsq“ ymj lwfhj nx xmj jswnhpji rj: wjxujhy fx lwt‘ yt xjrnsl ymj rtwsnsl: fsi ymjwj ‘fx t.jw ktw f xujhyjw?

Sample after 2500 fine-tuning iterations:

“He’s stood flowers.” Thom Egwene, the letting the gray and the carwled. “Well, I’m not get here?”

“You will sheep a stirred in a watchen, this cloak with on travely and his eyes. Whick the Two Rivers, after who ktart of the thought bed. The was already tower as if he wear cloak with a red granly and think that he hunsters hand seem how that bad time, then the different and both never fair, about that touching in the you are than and meant. Whate eogse up of Rand except of the other.

Prompting

1. Zero-shot vs Few-shot

Any encoded question in a not-so-popular cipher is an example of a question that ChatGPT cannot answer in a zero-shot manner, but can with a few-shot approach. Below is an example of an encoded question in the Rot-23 cipher asking the capital of France:

Zero-shot:

Question:

Texq fp qeb zxmfxi lc Coxkzb?,

Answer:

It seems like your question contains a typo or encoding issue. Could you please rephrase or clarify the question so I can assist you better?

Few-shot:

Question:

Example 1:

Q: Elt jxkv zlkqfkbkqp xob qebob fk qeb tloia?

A: Qebob xob pbsbk zlkqfkbkqp

Example 2:

Q: Texq fp qeb expqbpq ixka xkfjxi?

A: Qeb zebbqxe fp qeb expqbpq ixka xkfjxi

Example 3:

Q: Texq fp qeb ixodbpq mixkbq fk lro plixo pvpqbj?

A: Grmfqbo

Example 4:

Q: Tel mxfkqba qeb Jlkx Ifpx?

A: Iblkxoal ax Sfkzf

Example 5:

Q: Texq dxp al mixkqp xyploy colj qeb xqjlpmebob?

A: Zxoylk Aflufab

Answer the below question:

Q: Texq fp qeb zxmfxi lc Coxkzb?,

Answer:

A: Qeb zxmfxi lc Coxkzb fp Lmbkbbx

2. Ensembling and Majority Vote

I was unable to find an example of a prompt for which ChatGPT generates different answers and the majority vote converges to the right answer. Here are some of the things that I attempted:

I first tried to ask factual questions and explicitly told ChatGPT the temperature parameter to use when generating answers along with the number of answers. Unfortunately, this led to a situation where ChatGPT always generates the correct answer but the only variation is how verbose and flowery the language is for these answers based on the temperature parameter. Below is an example:

Question:

Use temperature=1.0 and generate 5 examples for following question: What is the capital of France?

Answer:

Q: What is the capital of France?

A: Paris, the romantic heart of France, where art, culture, and history collide.

Q: What is the capital of France?

A: The capital is Paris, a city that blends modern elegance with centuries of tradition.

The other 3 samples were similar to those above.

I then decided to ask ChatGPT a tricky question that could trick people into picking the wrong answer with the hope that with a higher temperature parameter, I might see some variance in the chosen answer. However, it continued to generate the correct answer in all cases for this too. Below is an example:

Question:

Use temperature of 1.0 and generate 5 answers for the following question:

"A bottle of water and a chocolate bar together cost \$1.10. The bottle of water costs \$1 more than the chocolate bar. How much does the chocolate bar cost?"

Answer:

Here are 5 highly creative responses with a temperature setting of 1.0:

A: The chocolate bar costs \$0.05, making the bottle of water \$1.05 — together, that's \$1.10!

A: It's a bit tricky, but the chocolate bar is \$0.05, and the water is \$1.05. Math magic!

Once again, the other 3 samples were similar to those above.

Finally, I tried asking a question that doesn't have an absolute correct answer but one that has a more probable right answer. Once again, the intuition was that since this only has a probable right answer, it might generate varied answers with a high-temperature parameter. Unfortunately, it converged to a single answer even in this situation. Below is an example:

Question:

Use temperature of 1.0 and generate 10 answers for the following question:

Which Team is more likely to win the match?

Team A: xG - 1.0 xGA - 1.2 Shots Taken: 1

Team B: xG - 1.2 xG - 1.0 Shots Taken: 2

Take the quality of shots into account as well where each shot's quality adds up to the total xG.

Answer:

Here are 10 creative responses, taking into account the shot quality and xG, with a temperature setting of 1.0:

A: Team B has a slight edge with more shots and a higher xG. They've got a better chance to convert those opportunities into a win.

A: With two shots and a higher xG, Team B seems more likely to score, even though Team A's single shot could be a high-quality chance.

A: Team B's two shots suggest they are creating more opportunities, and their higher xG gives them a slight advantage.

The remaining 7 samples were similar with the answer converging to Team B.

3. Rot-13

(a) What do you obtain if you ask a question like this zero-shot?

Question:

Jung vf gur pncvgny bs Senapr?

Answer:

The question you asked appears to be encoded using the ROT13 cipher. Let's decode it:

"Jung vf gur pncvgny bs Senapr?"

When decoded, this becomes:

"What is the capital of France?"

The answer is: Paris.

(b) Note: Since, ChatGPT was able to decode Rot-13 directly, I used the Rot-23 cipher instead for the follow-up questions. In the case of zero-shot, ChatGPT doesn't recognise that a cipher is being used. This is the response for the zero-shot question:

It seems like your question contains a typo or encoding issue. Could you

please rephrase or clarify the question so I can assist you better?

What do you obtain with a few-shot variant?

Question:

Example 1:

Q: Elt jxkv zlkqfkbkqp xob qebob fk qeb tloia?

A: Qebob xob pbsbk zlkqfkbkqp

Example 2:

Q: Texq fp qeb cxpqbpq ixka xkfjxi?

A: Qeb zebbqxe fp qeb cxpqbpq ixka xkfjxi

Example 3:

Q: Texq fp qeb ixodbpq mixkbq fk lro plixo pvpqbj?

A: Grmfqbo

Example 4:

Q: Tel mxfkqba qeb Jlkx Ifpx?

A: Iblkxoal ax Sfkzf

Example 5:

Q: Texq dxp al mixkqp xyploy colj qeb xqjlpmebob?

A: Zxoylk Aflufab

Answer the below question:

Q: Texq fp qeb zxmfxqi lc Coxkzb?,

Answer:

A: Qeb zxmfxqi lc Coxkzb fp Lmbkxbx

For Rot-23, asking the question in a few-shot manner leads to ChatGPT recognizing that this is a cipher and returning an answer. However, the answer returned is incorrect. When decoded, it reads:

The capital of France is Openea.

(c) Provide the model with additional instructions. What can you obtain?

Question:

Example 1:

Q: Elt jxkv zlkqfkbkqp xob qebob fk qeb tloia?

A: Qebob xob pbsbk zlkqfkbkqp

Example 2:

Q: Texq fp qeb expqbpq ixka xkfjxi?

A: Qeb zebbqxe fp qeb expqbpq ixka xkfjxi

Example 3:

Q: Texq fp qeb ixodbpq mixkbq fk lro plixo pvpqbj?

A: Grmfqbo

Example 4:

Q: Tel mxfkqba qeb Jlkx Ifpx?

A: Iblkxoal ax Sfkzf

Example 5:

Q: Texq dxp al mixkqp xyploy colj qeb xqjlpmebob?

A: Zxoylk Aflufab

The five examples listed above consist of a question and it's corresponding answer. Both the question and the answer are encoded using some simple rotation cipher. I want you to go through the five examples and deduce the cipher that is being used.

After that, I want you to decode the question given below using the same cipher, find the answer to it and return that answer back.

Answer the below question:

Q: Texq fp qeb zxmfxi lc Coxkzb?,

Answer:

To solve this problem, we'll first deduce the cipher used in the examples, then decode the provided question, find the answer, and finally encode the answer using the same cipher.

Cipher Deduction

Upon examining the examples, we notice that each letter in the text is shifted by a consistent number of positions in the alphabet.

Let's analyze Example 1:

Encoded Question: "Elt jxkv zlkqfkbkqp xob qebob fk qeb tloia?"
We'll try shifting each letter forward by 3 positions in the alphabet.
For the word "Elt":

$E (5) + 3 = H (8)$
 $L (12) + 3 = O (15)$
 $T (20) + 3 = W (23)$
So, "Elt" deciphers to "How".

Applying this shift to the rest of the text confirms that a Caesar cipher with a shift of +3 is used.

Decoding the Question

Now, let's decode the provided question using the shift of +3.

Encoded Question: "Texq fp qeb zxmfxqi lc Coxkzb?"

Let's shift each letter forward by 3 positions:

$T \rightarrow W$ $E \rightarrow H$ $X \rightarrow A$ (since $X \rightarrow A$, wrapping around the alphabet) $Q \rightarrow T$

So, "Texq" deciphers to "What".

Continuing with the rest:

$F \rightarrow I$ $P \rightarrow S$
"Fp" becomes "Is".

$Q \rightarrow T$ $E \rightarrow H$ $B \rightarrow E$
"Qeb" becomes "The".

$Z \rightarrow C$ $X \rightarrow A$ $M \rightarrow P$ $F \rightarrow I$ $Q \rightarrow T$ $X \rightarrow A$ $I \rightarrow L$
"Zxmfxqi" becomes "Capital".

$L \rightarrow O$ $C \rightarrow F$
"Lc" becomes "Of".

$C \rightarrow F$ $O \rightarrow R$ $X \rightarrow A$ $K \rightarrow N$ $Z \rightarrow C$ $B \rightarrow E$
"Coxkzb" becomes "France".

Decoded Question: "What is the capital of France?"

Answering the Question

The capital of France is Paris.

Adding some additional instructions to the prompt that this is a rotation cipher that you must decode with the help of the examples helps ChatGPT deduce the cipher and return the answer in a human-readable format instead of an incorrectly encoded answer.

- (d) Find a strategy to ultimately produce the correct answer to an encoded geographic (or other) question like this one.

The strategy mentioned in the previous question of adding some additional instructions to the prompt that this is a rotation cipher that you must decode with the help of the examples helps ChatGPT deduce the cipher and return the answer in a human-readable format instead of an incorrectly encoded answer.

This seems to work for other ciphers and questions as well.