Neural networks assignment 5 report
Sarah Hosseini 400222026

In this assignment, we implemented a transformer based model to generate Persian text, leveraging the rich and diverse content available in the Persian Wikipedia corpus. Throughout this report, we will discuss the dataset preprocessing steps, model configuration, training methodology, and evaluation metrics used to assess the performance of our Transformer-based Persian text generator. Moreover, we will present key findings, challenges encountered, and potential areas for future improvements in our quest to develop a state-of-the-art language model for Persian text generation.

Dataset and preprocessing

I used the dataset available at 'this kaggle link'. It contains all wikipedia articles to the date of Mordad 1399. The data was 7 .txt files. Since all that data would have taken too much time and computation resources, I used only one of those files. After loading the file into my notebook, I did some exploratory analysis on the data: this data had a length of 52303455 characters and approximately 9768905 words. The dataset had articles that were separated by the keyword "عنوان مقاله" followed by the article's title. Since this much data was still too large for my small model, I cut the first 1/10 portion of the text data and used only that. This resulted in minimizing my dataset to a corpus of length 5230345 (in characters) and 1004072 words. Some of data that contained these characters looked like this:

ها ندارند و به همین دلیل، نحوه تکثیر آن ها هنوز برای انسان ها نامشخص است. اجساد تایتان ها بلافاصله ...
پس از مرگ به سرعت به بخار تبدیل می شود. دمای بدن تایتان ها بالاست. برخی از تایتان ها را غیرعادی (奇
行種 Kikō-shū) می نامند چون برخلاف دیگر تایتان ها عمل می کنند.

تایتان غول پیکر(超大型巨人 Chō-ōgata Kyojin) (Colossal Titan)
... بزرگترین تایتانی که تا به حال دیده شده است. (با طولی نزدیک به ۶۰ متر) این تایتان

Then, I printed the set of all the characters that were used and found a mixture of Japanese, English, Chinese, Hebrew,... characters. There were totally 582 different characters used. Since I want my model to learn solely Persian, I proceeded to delete all those characters and replace them by spaces. This caused the dataset to contain only 87 characters listed below (plus ‘newline’ and ‘space’):

0123456789…
ءآأؤإئابةتثجحخدذرزسشصضطظعغفقكلمنهوىي٠١٢٣٤٥٦٧٨٩٪،,أپځچڕژکڪگۀیه۰۱۲۳۴۵۶۷۸۹

Then, I divided the dataset into train and validation sets with a ratio of 9 to 1. I used the Dataset class of pytorch to build my own dataset class. My dataset tokenizes the data at character level, then using a dictionary maps them to numbers between 0 and 86. My dataset class also has a getter which returns two sequences of characters: one as the input to my model and the other as the expected output. So if it has to return the zeroth element of my dataset and my data begins like this:
5,60,3,4,51,35,52,86,9,0,...
It will return two sequences: ‘5,60,3,4’ and ‘60,3,4,51’. It means that if the model saw 5, it should return 60. If it saw ‘5,60’, it should return 3. If it saw 5,60,3 it should return 51 and so on. Also, the sequence length is a hyperparameter that the dataset has to know at the initialization step.

Model

Our model is a simple transformer (the decoder part of the GPT model) that is built upon the self attention mechanism with query, key and value system. Our LLM’s forward() method is the core method for training the model. It takes in token indices and, optionally, target labels targets.
If no targets are provided, it means we want to only generate text and loss will be none. But when we have a target, we are predicting.
The model first applies token and positional embeddings to the input and passes the resulting embeddings through the Transformer blocks. The resulting output is passed through the final layer normalization and linear layer, resulting in logits for each token. The cross-entropy loss is computed if targets are provided. The method returns the logits and the loss.
When doing text generation, the ‘generate’ method generates text given an initial context encoded in ‘idx’. It iteratively samples the next token for a specified number of ‘max_new_tokens’. The method crops the last ‘seq_len’ tokens from

the context, generates logits, applies softmax to obtain probabilities, and samples the next token. The sampled token is appended to the context, and the process is repeated for the desired number of tokens. The method returns the generated token indices.

Each of those ‘Blocks’ compose of one multihead self-attention component followed by a feed forward component and each of the components also add the input to their output (skip connection).

This model has 58199 parameters.

Training

I used batches with size 32 and sequences of length 64 to train my model. I used AdamW optimizer and a learning rate of 1e-3 with 10 iterations and the model got to see 500 batches in each iteration. The training loss and validation loss were measured and printed after every 3 iterations.
The loss started at 4.6 and after 10 epochs, decreased to 1.86 for train and 2.24 for validation. (the higher value for validation indicates some overfitting).
The model generated this sample:

```
اثضالی حراط یاص رابا کند ندر بشه شد سیاد و که اثثته  زبا شخیتر شورگی او شسه راندین ۱۹ خورض  به دالایت او در دهمت  آو
صرب امننیخی و وجه دی۳ ا حسهر حرامم موبلا ۲۷۵ میل بتر مرحدی  درنشک در به جور حزن محصد    نیاز  را زشدماران علام منا
خوام  شوده اننست نگرامعی نوزه ارست  هانیهانگیر زمنداد  هست ایده  پلاقه تجار های درآن در از گرف خواله ژوازی عن سیل تو
جمتات دادامنی به میروجه ای کردگرگی  شدیربان شباعال می شون اصه تند  شاط که بانشد



عنوان مقال  اساس کلادی
این شدندهای ۱۹۶ یک که طول آن 2۴۲۱
```

This doesn’t have many recognizable Persian words. But the words ‘عنوان مقاله’ which were repeated a lot throughout the corpus have been learnt and the model has learned to produce some string that has the same formatting as my corpus. This means that I should have removed the repetitive phrase (عنوان مقاله) and the unnecessary spaces and new lines so my model would have been able to learn Persian, not the formatting of the dataset.

I continued training the model for another 90 epochs and the loss went down to 1.54 for train and 1.88 for validation.
Now the generated text is like this:

```
عنوان مقاله  اچ ام اس کانفت  ۱۸۴

اچ ام اس ارسر  آب ۱۷۸  یک کشتی است که طول آن می باشد



عنوان مقاله  رلیسک

مئیس اف الیجی با  پی استون   تبکی فضایه  سرل نکه نزد آمبیا اختلار تا بادولت یک  محسوبل از به سازنه از تهرایی و آینها  ترکتان
متولیسونایژ بنای اسلی  دادگاه های به کترو سخن در سال ۱۹۴۴ک از چایستان موارسه گبرد
از سرزمی برگشتن  تبرر به اسراتکونی آبهنس  سنگ تیکان آن   پزیکلوس کرد
کلیلا  های   گروهای حردومه  ژورت به سند با ئرام است که بیمارساسی رساندگی در نظیم داتثرت می خوانوند و می یک ویتر که
حسائوت  آرگال ها تبرای گرتازه می باشد  گره که کلاه هری در سنای ایالات متحده آمریکا بود
```

Which shows more coherent words that are indeed persian. After another 30 iterations, we losses of 1.50 and 1.87. The text:

```
پی ک سناتور سابق عضو حزب دموکرات ایالت اممرده ای را یکملز ژوئی چشمنی های هنگزیدرایی محلات است  تراک در ۱۳۰  سال
۱۸۱۲ میلادی بود که طول آن بود  این کشتی در سال ۱۸۴۴ ساخته طول آن می بایلات میلاد دهد  سختت وجود که طول آن بود



عنوان مقاله  فوات وی رسوم عمل اب در سپس امورد  ۵۹۹ هجهان سازی علیف او تاوئیه می داند

فنای مروش تحقیقات محمدعه خداده خنط محیط نیابد به نوع مصرف برای پوس نشای عبدری خم کابکرنان علدان قمومت عنقلاع هموا به
رای آی ستانیما به عنشی یک در است ارمرد و برای است

دوم دانشگیت طبق دمان یشتری به طول آن بود  این کشتی در سال ۱۹۱۹ ساخته شد
```

At the end, I printed some of the expected sequences against the predicted sequences:

```
expected:  دانشجویان سفیدپوست دانشگاه فیسک را دوره کردند و با یک چمدان آنقد
predicted:  دانشجویان سفیدپوست دانشگاه فیسک را دوره کردند و با یک چمدان آنقدون هستند   فاست  عموم ترین لمه ۱۱۸۲۰ ژنده
دهند که هید و را جزازا
expected:  کوب توسط پلیس و فعالیت های تروریستی کوکلوس کلان ها در بیرمنگام
```

predicted: کوب توسط پلیس و فعالیت های تروریستی کوکلوس کلان ها در بیرمنگام پاکس سید است استفاده از گیانی کتاب است که
معمارش کشور یورید

د
expected: تیجه یکسان به دست آمده نشان می دهد هر گراف مسطح با درجه متناهی ت
predicted: تیجه یکسان به دست آمده نشان می دهد هر گراف مسطح با درجه متناهی توم دادن این مسابق شده اند کردن   باقی حافظ
میرده اند بر موسط نا

Also, the mean Rouge1 and Rouge2 and perplexity measurements were calculated for 50 batches of size 32 of the validation set.
Perplexity measures how well a model predicts the next token given the previous tokens. Lower perplexity values indicate better model performance. A perplexity of 3.7 is generally considered good, as it suggests that the model can effectively predict the next token in a given context.
ROUGE-1 measures the overlap of unigrams (single words), while ROUGE-2 measures the overlap of bigrams (pairs of words). The scores range from 0 to 1, where higher values indicate better performance.
In my case, the low ROUGE-1 (0.01) and ROUGE-2 (0.003) scores suggest that the model's generated texts have limited overlap with the actual reference texts. This indicates that the model is not performing well on the predicting task.
The best way to make our model better would be to improve the preprocessing and make our text more representative of the Persian language, and also using more of the wikipedia dataset will definitely improve our model. Making our model use more transformer block and more parameters will be helpful too. And tokenizing at word level and encoding the characters in a more meaningful way like word2vec will be helpful too.