
Week 3 Assignment [4 Points]

Generative AI

Saarland University – Winter Semester 2024/25

Submission due: 7 November 2024 by 6pm CET

genai-w24-tutors

genai-w24-tutors@mpi-sws.org

1 Reading Assignment

This week's reading assignment comprises of the following material:

- (R.1) Chapters 10.0 (i.e., introduction), 10.1, 12.0 (i.e., introduction), 12.1, 12.4, and 14.3 from the book by Jurafsky and Martin [1]. The book is available freely online at the following link: https://web.stanford.edu/~jurafsky/slp3/ed3bookaug20_2024.pdf.
- (R.2) [GPT-3] Language Models are Few-Shot Learners [2]. Paper link: <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>.

[OPTIONAL] Below, we are providing additional material covering content relevant to the lecture:

- Chapters 9 (whole chapter) and 10.2 from the book by Jurafsky and Martin [1].
- [GPT-1] Improving Language Understanding by Generative Pre-Training [3]. Paper link: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.
- [GPT-2] Language Models are Unsupervised Multitask Learners [4]. Paper link: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.
- [GPT-3] Longer version of the NeurIPS'20 paper [2] available on arXiv. Paper link: <https://arxiv.org/pdf/2005.14165>.
- Chain-of-Thought Prompting Elicits Reasoning in Large Language Models [5]. Paper link: https://openreview.net/pdf?id=_VjQlMeSB_J.

2 Exercise Assignment

This week's exercise assignment aims to give you insights into the transformer-based language model architecture and how various prompting strategies can affect the output of a large language model. These questions should be answered in the PDF submission file – see instructions in Section 4.

- (E.1) Consider the layer of attention heads in Week 3's lecture slides #18–19. The input to the attention layer is $N - 1$ embedding vectors of dimension d denoted by $\mathbf{e}_{t-N+1}, \dots, \mathbf{e}_{t-2}, \mathbf{e}_{t-1}$. One attention head uses query matrix W^q (size $d_k \times d$), key matrix W^k (size $d_k \times d$), and value matrix W^v (size $d_v \times d$). There are η_a attention heads in the attention layer. Note that d , d_k , and d_v could be different from each other. Write down detailed steps to compute the attention vector \mathbf{a} as output from the attention layer.
- (E.2) Similar to the above question, again consider the layer of attention heads in Week 3's lecture slides #18–19. There are η_a attention heads in the attention layer. Note that d , d_k , and d_v could be different from each other. Write down detailed steps to find the total number of parameters in this multi-head attention layer.

(E.3) Consider the transformer block in Week 3’s lecture slides #22–29 and described below. Write down detailed steps to find the total number of parameters in this transformer block.

- **[Input vectors]** The input to the transformer block is $N - 1$ vectors of dimension d .
- **[Attention layer]** These $N - 1$ vectors are fed into the attention layer. The attention layer has η_a heads, where each attention head uses query matrix W^q (size $d_k \times d$), key matrix W^k (size $d_k \times d$), and value matrix W^v (size $d_v \times d$). Note that d , d_k , and d_v could be different from each other. The output from the attention layer is a vector of dimension $\eta_a d_v$ which is obtained by concatenating the output vectors from each of the attention heads.
- **[Attention projection, sum, norm]** This step takes the output vector from the attention layer (size $\eta_a d_v$), projects it to a vector of size d using a matrix W^{AProj} (size $d \times \eta_a d_v$), adds the residual, and applies layer normalization. Note that the layer normalization has two learnable parameters, namely γ^{ANorm} and β^{ANorm} .
- **[Feedforward layer]** The feedforward (hidden) layer takes as input a vector of size d obtained from the previous step and has d_h neural units. The output is a vector of size d_h .
- **[Feedforward projection, sum, norm]** This step takes the output vector from the feedforward layer (size d_h), projects it to a vector of size d using a matrix W^{FFProj} (size $d \times d_h$), adds a bias term vector \mathbf{b}^{FFProj} , adds the residual, and applies layer normalization. Note that the layer normalization has two learnable parameters, namely γ^{FFNorm} and β^{FFNorm} .
- **[Output vector]** The output of the transformer block is one vector of dimension d .

(E.4) Consider the transformer architecture in Week 3’s lecture slides #31–33. This transformer architecture has L layers of transformer blocks (i.e., number of transformer blocks in a column). Let’s denote the number of parameters in one transformer block by B . The input to this transformer architecture is given by $N - 1$ words denoted by $\mathbf{x}_{t-N+1}, \dots, \mathbf{x}_{t-2}, \mathbf{x}_{t-1}$ and represented as one-hot encoded vectors of size $|V|$. These one-hot encoded vectors are first converted to embedding vectors of size d using the embedding matrix E (size $d \times |V|$) before being processed by the transformer blocks. The output from these transformer blocks is then converted back to a vector of size $|V|$ using unembedding matrix U (size $d \times |V|$) – note that the weights are shared between the embedding matrix E and the unembedding matrix U . Write down the total number of parameters in this transformer architecture in terms of the above-mentioned variables, namely L , B , N , $|V|$, and d .

(E.5) This question involves comparing zero-shot vs. few-shot prompting strategies for a translation task (see Figure 1). The task involves translating a sentence from Old Romanian to English – the correctly translated sentence should capture a welcoming sentiment related to inviting someone to eat. The generated output for the task would be reported as “correct” if the translation sentence roughly captures this invitation.

You will use OpenAI’s *GPT-4o-mini model* for this question. Importantly, you should access GPT-4o-mini through <https://chatgpt.com/> using an *incognito browser window* – the incognito mode will ensure that your conversation history does not affect the results. Moreover, you will have to *refresh the page after every query* to remove the conversation history. More concretely, report results for the following two strategies:

- Zero-shot: Perform the translation task using the zero-shot prompt shown in Figure 1a. To account for potential stochasticity in the model’s output, perform the translation task three times (refreshing the page every time) and report how many times the generated output is “correct”.
- Few-shot: Perform the translation task using the few-shot prompt shown in Figure 1b. To account for potential stochasticity in the model’s output, perform the translation task three times (refreshing the page every time) and report how many times the generated output is “correct”.

In your submission for this question, report the number of times you got the “correct” output for the two prompting strategies. Provide a brief discussion on how the two prompting strategies compare for this translation task.

(E.6) This question involves comparing basic vs. chain-of-thought (CoT) prompting strategies for three reasoning tasks, referred to as Task-A, Task-B, and Task-C (see Figure 2). The

generated output for a task would be reported as correct if the final answer matches the expected answer: 990 for Task-A, 1,080 for Task-B, and 520 for Task-C.

You will use OpenAI's *GPT-4o-mini model* for this question. As in the previous question, you should access GPT-4o-mini through <https://chatgpt.com/> using an *incognito browser window* – the incognito mode will ensure that your conversation history does not affect the results. Moreover, you will have to *refresh the page after every query* to remove the conversation history. More concretely, report results for the following two strategies:

- Basic prompting: Perform each of the three tasks using the basic prompts shown in Figures 2a, 2c, and 2e. To account for potential stochasticity in the model's output, perform each task three times (refreshing the page every time) and report how many times the generated output is correct for each task.
- CoT prompting: Perform each of the three tasks using the CoT prompts shown in Figures 2b, 2d, and 2f. To account for potential stochasticity in the model's output, perform each task three times (refreshing the page every time) and report how many times the generated output is correct for each task.

In your submission for this question, report for each task the number of times you got the correct output for the two prompting strategies. Provide a brief discussion on how the two prompting strategies compare for these reasoning tasks.

3 Implementation Assignment

This week's implementation assignment is designed so that you can get familiarity with running pre-trained large language models (LLMs) in Google's Colab environment. We have provided the following files in `week3_implementation.zip`:

- `llm_completion.py`: An implementation that automatically downloads and makes use of an LLM from Hugging Face (<https://huggingface.co/>) in order to generate completions for given prefixes. Note that the model gets downloaded to the Colab environment, not to your local machine. The script reads an input file containing text prefixes, generates completions for each prefix, and outputs its results in a new file. The script is capable of using GPU resources from Colab when available – see Section 5.2 for more details on how to use GPU resources.
- `input_shakespeare_prefixes.txt`: This file contains a set of text prefixes from Shakespeare's works, which the LLM will use as prompts for generating completions.

As part of this assignment, you will simply execute the provided code without implementing any new functionality. More concretely, in the `__main__` function of `llm_completion.py`, the script will read the input file `input_shakespeare_prefixes.txt`, generate completions using the LLM, and write the generated completions to `output_shakespeare_completions_{MODEL_NAME}.txt`. Once the code is executed, you can review the generated output.

- (I.1) In the `__main__` function of `llm_completion.py`, set the `MODEL_NAME` to `openai-community/gpt2-xl`¹ in order to use GPT-2 (XL version with 1.5 billion parameters) as the LLM, then run the script. The completions will be written to `output_shakespeare_completions_gpt2-xl.txt`. This `.txt` file should be included as part of the ZIP submission file – see instructions in Section 4.
- (I.2) In the `__main__` function of `llm_completion.py`, set the `MODEL_NAME` to `microsoft/phi-1_5`² in order to use Phi-1.5 as the LLM, then run the script. The completions will be written to `output_shakespeare_completions_phi-1_5.txt`. This `.txt` file should be included as part of the ZIP submission file – see instructions in Section 4.
- (I.3) Provide a qualitative comparison of how the generated completions vary in terms of quality and characteristics for GPT-2 (XL version with 1.5 billion parameters) and Phi-1.5. You can answer this question in a few sentences and no detailed analysis is required. This should be answered in the PDF submission file – see instructions in Section 4.

¹You can read more details about this model at <https://huggingface.co/openai-community/gpt2-xl>.

²You can read more details about this model at https://huggingface.co/microsoft/phi-1_5.

Translate this sentence from Old Romanian to English:

Romanian: Ie' de-nbucă oarișce!

English:

(a) Prompt for zero-shot translation (you can simply copy-and-paste this prompt).

Translate this sentence from Old Romanian to English:

Romanian: Ie și be' de-acolea!

English: Take and drink from there!

Romanian: Atuncé poati ca să nădăjduiască oarișce spre mila domnească.

English: Then [he] can have some hope for the mercy of his lord.

Romanian: După o zi obositoare, pădurarul se întoarce la cabană și începe a-nbucă bunătățile pregătite de mama sa.

English: After a tiring day, the forester returns to the cabin and begins to eat the food prepared by his mother.

Romanian: Ie' de-nbucă oarișce!

English:

(b) Prompt for few-shot translation (you can simply copy-and-paste this prompt).

Figure 1: **E.5:** Prompts for zero-shot vs. few-shot translation.

Tina makes \$18.00 an hour. If she works more than 8 hours per shift, she is eligible for overtime, which is paid by your hourly wage + 1/2 your hourly wage. If she works 10 hours every day for 5 days, how much money does she make? Just give an answer.

(a) Reasoning Task-A: Basic prompt (you can simply copy-and-paste this prompt).

Tina makes \$18.00 an hour. If she works more than 8 hours per shift, she is eligible for overtime, which is paid by your hourly wage + 1/2 your hourly wage. If she works 10 hours every day for 5 days, how much money does she make? Think step by step, then give an answer.

(b) Reasoning Task-A: CoT prompt (you can simply copy-and-paste this prompt).

Each bird eats 12 beetles per day, each snake eats 3 birds per day, and each jaguar eats 5 snakes per day. If there are 6 jaguars in a forest, how many beetles are eaten each day? Just give an answer.

(c) Reasoning Task-B: Basic prompt (you can simply copy-and-paste this prompt).

Each bird eats 12 beetles per day, each snake eats 3 birds per day, and each jaguar eats 5 snakes per day. If there are 6 jaguars in a forest, how many beetles are eaten each day? Think step by step, then give an answer.

(d) Reasoning Task-B: CoT prompt (you can simply copy-and-paste this prompt).

Tara has been planning to buy a laptop which costs \$1000. A computer shop accepts payment in installments of \$65 per month provided that a 20% down payment is made. If Tara wants to pay an additional \$20 for the down payment, how much will her balance be after paying for 4 months? Just give an answer.

(e) Reasoning Task-C: Basic prompt (you can simply copy-and-paste this prompt).

Tara has been planning to buy a laptop which costs \$1000. A computer shop accepts payment in installments of \$65 per month provided that a 20% down payment is made. If Tara wants to pay an additional \$20 for the down payment, how much will her balance be after paying for 4 months? Think step by step, then give an answer.

(f) Reasoning Task-C: CoT prompt (you can simply copy-and-paste this prompt).

Figure 2: **E.6:** Prompts for basic vs. CoT prompting on three reasoning tasks [6].

4 Submission Instructions

Please submit the following file:

- **<Lastname>_<Matriculation>_week3.pdf**. This PDF file will report answers to the following questions: E.1, E.2, E.3, E.4, E.5, E.6, I.3. The PDF file should be generated in LaTeX using NeurIPS 2024 style files; see further formatting instructions in Section 5.1. The PDF file size should **not exceed 1mb**. Please use the following naming convention: Replace <Lastname> and <Matriculation> with your respective Lastname and Matriculation number (e.g., Singla_1234567_week3.pdf).
- **<Lastname>_<Matriculation>_week3.zip**. This ZIP file will report answers to the following questions: I.1, I.2. Importantly, the ZIP file should unzip to a folder named **<Lastname>_<Matriculation>_week3**. Inside this folder, include two .txt files containing the generated completions. The ZIP file size should **not exceed 1mb**. Replace <Lastname> and <Matriculation> with your respective Lastname and Matriculation number (e.g., Singla_1234567_week3.zip).

5 Technical Instructions

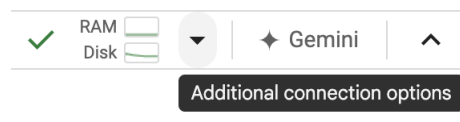
5.1 Instructions for Preparing PDFs

Refer to Week 1 assignment.

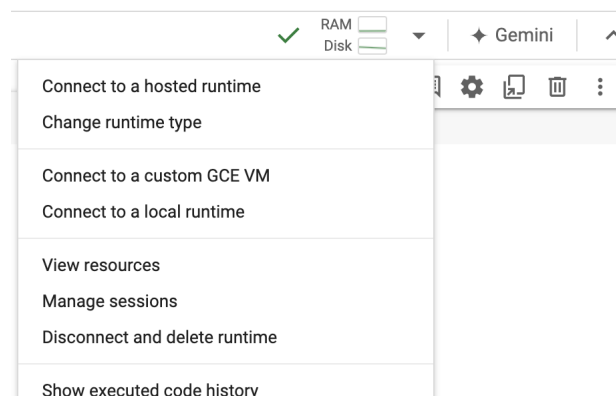
5.2 Instructions for Implementation

To use CPU-based hardware resources in Google Colab for this assignment, you can refer to the instructions provided in Week 1 assignment. However, CPU-based hardware would have slower inference time. To speed up the inference time, below we are providing instructions on how to connect to GPU-based hardware resources in Google Colab. Note that GPU-based resources may not always be available at a given time – in this case, you can try later or simply use CPU-based resources.

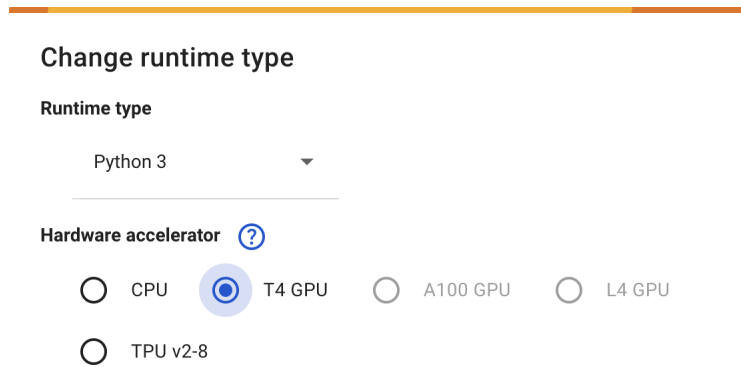
- (1) Ensure that you are connected to a runtime session. Then, get drop-down menu with additional connection options as shown below.



- (2) Click on “Change runtime type” in the drop-down menu.



- (3) Select T4 GPU as hardware accelerator.



Change runtime type

Runtime type

Python 3 ▼

Hardware accelerator ?

☐ CPU ☒ T4 GPU ☐ A100 GPU ☐ L4 GPU

☐ TPU v2-8

- (4) Check that you are connected to GPU-based resources as shown below.



- (5) In case you make changes to your files within the editor, ensure that these files are saved elsewhere. The uploaded files will be deleted automatically once a runtime session is changed or terminated.

References

- [1] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition, 2024. Online manuscript released August 20, 2024; <https://web.stanford.edu/~jurafsky/slp3/>.
- [2] Tom B. Brown et al. Language Models are Few-Shot Learners. In *NeurIPS*, 2020.
- [3] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.
- [4] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf, 2019.
- [5] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS*, 2022.
- [6] Karl Cobbe et al. Training Verifiers to Solve Math Word Problems. *CoRR*, abs/2110.14168, 2021.