

## Assignment 1: Text Generation and Language Modeling

Date: May 20, 2025 Due : June 5, 2025

### PREFACE

This is the first series of assignments for the course *ECE1508: Deep Generative Models*. The exercises are aimed to review the language modeling studied in Chapter 1. Please adhere to **Code of Honor** outlined on the course page. Below, you can find the information about the contents, as well as instructions on how to submit them. Please read them *carefully*.

**GENERAL INFORMATION** The assignments are given in two sections. In the first section, you have written questions that can be answered in words or by derivation. The questions are consistent with the material of Chapter 1 and you do not need any further resources to answer them. The second section includes the programming assignments. In the case that a question is unclear or there is any flaws, please contact over Piazza. Also, in case that any particular assumption is required to solve the problem, feel free to consider the assumption and state it in your solution. The total mark of the assignments is **100 points** with total mark of written questions adds to **50 points** and the total mark of the programming assignments adds to **50 points**.

**HOW TO SUBMIT** A notebook has been provided with starter code that you can complete. The submission is carried out through the Crowdmark. Please submit the answer of each question **separately**, following **the steps below**. Please note that *failure to meet the formatting can lead to mark deduction*.

1. For Written Questions, you can submit handwritten or typed answers as a `.pdf`, `.jpg` or `.png` file.
2. For Programming Questions, the answer should **complete the Python Notebook** shared with this assignment. For *each question*, please print the corresponding part of the notebook as a `.pdf` file and upload it in the corresponding field on Crowdmark. Your uploaded `.pdf` should contain **all figures, diagrams and codes requested** in the question.
3. The completed Notebook, i.e., the `.ipynb` file, including all the codes and the outputs should also be submitted as the attachment to the last item on Crowdmark.

When submitting your notebook, please pay attention to the following points:

1. The file should be renamed as `<YourLastname>_<YourFirstname>_Assgn1.ipynb`
2. Please make sure to name the files with the name that is displayed on the Quercus account

The deadline for your submission is on **June 5, 2025 at 11:59 PM**.

- You can delay up to two days, i.e., until **June 7, 2025 at 11:59 PM**. After this extended deadline no submission is accepted.
- For each day of delay, 10% of the mark after grading is deducted.

Please submit your assignment **only through Crowdmark, and not by email**.

# 1 WRITTEN EXERCISES

**QUESTION 1** [20 Points] (**Dummy Language**) Consider a dummy language with only two letters A and B. This language describes a Markov process of order 1, i.e., the distribution of each letter in the text is *fully described* by its previous letter, with the following *true* conditional distribution

$$P(x|A) = \begin{cases} 0.5 & x = A \\ 0.5 & x = B \end{cases} \quad P(x|B) = \begin{cases} 0.25 & x = A \\ 0.75 & x = B \end{cases}$$

We intend to train a simple bi-gram model on the following sample corpus of this language.

ABABAABBABBBBAAABBBB

1. Specify the dataset  $\mathbb{D}$  that consists of *all* possible samples in the given corpus.  
**Hint:** Remember that this language is a rank-one Markov process, so we only need samples of length 2.
2. Evaluate the empirical language distribution  $\hat{P}(x_t|x_{t-1})$  using the dataset  $\mathbb{D}$ . Does this distribution matches the true distribution  $P(x_t|x_{t-1})$ ?
3. Specify a tokenization for this language. Tokenize the corpus using the proposed tokenization.
4. Specify the basic bi-gram model with two embeddings (similar to the one we had in the lecture). Write the explicit expression for the probability distribution  $P_{\mathbf{E}}(x_t|x_{t-1})$  specified by this model, where  $\mathbf{E}$  denotes the  $2 \times 2$  embedding matrix, i.e.,

$$\mathbf{E} = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

for  $a, b, c, d \in \mathbb{R}$  that are to be learned.

5. Find the average log-likelihood of the model computed over the *whole* dataset  $\mathbb{D}$  as a function of  $\mathbf{E}$ , i.e., find

$$\mathcal{L}(\mathbf{E}) = \frac{1}{B} \sum_{i=1}^B \log P_{\mathbf{w}}(x_t^i|x_{t-1}^i),$$

where  $(x_{t-1}^i, x_t^i) \in \mathbb{D}$  is the  $i$ -th sample, and  $B$  is the number of samples in  $\mathbb{D}$ .

6. Train the model by maximizing  $\mathcal{L}(\mathbf{E})$  over  $\mathbf{E}$ .  
**Attention:** You can easily solve it by hand. To this end, set partial derivatives w.r.t. entries of  $\mathbf{E}$  to zero.
7. Specify the distribution learned after training the bi-gram model, and compare it against the true distribution  $P(x_t|x_{t-1})$  and the empirical one  $\hat{P}(x_t|x_{t-1})$ . What do you conclude?

**QUESTION 2** [10 Points] (**Divergence**) Two distributions can be compared using a *divergence* metric: when two distributions are the same, their divergence is zero and when they are different, the divergence is positive. The larger the divergence, the more the difference between the distributions. In this question, we work with *Kullback-Leibler (KL) divergence* considering the setting in Question 1.

**KL Divergence:** Assume that we have two conditional distributions  $P(x_t|x_{t-1})$  and  $Q(x_t|x_{t-1})$ . The KL divergence between these two distributions is

$$D_{\text{KL}}(P\|Q) = \sum_{x_{t-1}} P(x_{t-1}) \sum_{x_t} P(x_t|x_{t-1}) \log \frac{P(x_t|x_{t-1})}{Q(x_t|x_{t-1})}$$

Note that in addition to conditional distributions, we need the *marginal* distribution  $P(x_{t-1})$ .

1. Compute the KL divergence between the true language distribution  $P(x_t|x_{t-1})$  and the distribution  $P_E(x_t|x_{t-1})$  given by the trained model in Question 1, i.e.,  $D_{\text{KL}}(P\|P_E)$ . Assume  $P(x_{t-1})$  is *uniform*.
2. Compute the KL divergence between the *empirical* language distribution  $\hat{P}(x_t|x_{t-1})$  and the trained bi-gram model  $P_E(x_t|x_{t-1})$  in Question 1, i.e.,  $D_{\text{KL}}(\hat{P}\|P_E)$ . If needed, evaluate marginal distribution  $\hat{P}(x_{t-1})$  from the corpus in Question 1 *empirically*.
3. Compare the results in Parts 1 and 2. What do you conclude?

**KL Divergence Estimate:** We can estimate the KL divergence from a batch of  $B$  samples as

$$\hat{D}_{\text{KL}}(P\|Q) = \frac{1}{B} \sum_{i=1}^B \log \frac{P(x_t^i|x_{t-1}^i)}{Q(x_t^i|x_{t-1}^i)}$$

4. Find the estimate  $\hat{D}_{\text{KL}}(P\|P_E)$  from dataset  $\mathbb{D}$  in Question 1 as a function of  $\mathbf{E}$ .
5. Compare the result of Part 3 to the average log-likelihood  $\mathcal{L}(\mathbf{E})$  determined in Question 1. What do you conclude?

**QUESTION 3 [10 Points] (Temperature)** As seen in the course, language models (LMs) compute a distribution from the context, i.e., the logits of the last layer, using the Softmax. A classical method to control this distribution is to introduce the hyperparameter *temperature*: this parameter controls the level of randomness, when we sample. To understand the functionality of temperature, let us consider a simple case.

Let  $\mathbf{z} \in \mathbb{R}^I$  be the logits of the last layer in a LM. We compute the probability of next token as

$$\mathbf{p} = \text{Softmax}\left(\frac{\mathbf{z}}{\theta}\right)$$

where  $\theta > 0$  is a real number referred to as *temperature*.

1. Explain what happens to the probability vector  $\mathbf{p}$  when
  - (a)  $\theta \rightarrow \infty$
  - (b)  $\theta \rightarrow 0$
2. Considering these two extreme cases, explain how  $\theta$  controls the randomness of sample generated by sampling from  $\mathbf{p}$ .

**QUESTION 4 [10 Points] (Retrieval Augmented Generation)** Retrieval-augmented generation (RAG) is a framework that combines retrieval from an external knowledge source, e.g., a document, with generative power of a language model (LM) [2]. The idea in RAG is that instead of relying solely on knowledge learned by the LM through training, we can use relevant documents retrieved at inference time to guide the generation process. In this question, we develop a basic understanding of RAG.

Let  $x_{1:t}$  denote a *prompt or input query* to a LM,  $D$  be a document retrieved from a large corpus  $\mathcal{D}$ , and  $y_{1:L}$  the *desired output*, e.g., an answer to the question in  $x_{1:t}$ . Our goal is to sample a response from the task-specific distribution  $Q(y_{1:L}|x_{1:t})$ . Suppose we get access to the following distributions through learning procedure

- $P_\alpha(D|x_{1:t})$  learned<sup>1</sup> for each document  $D \in \mathcal{D}$ , where  $\mathcal{D}$  denotes the corpus of documents that we have access to, and
- $P_{\mathbf{w}}(y_{1:L}|D, x_{1:t})$  which is learned for each  $D, x_{1:t}$  by the LM through fine-tuning or prompt design.

<sup>1</sup>This model is parameterized by some parameters in  $\alpha$ . In practice, we call this model *retriever*.

Answer to the following items.

1. Intuitively explain what the models  $P_{\alpha}(D|x_{1:t})$  and  $P_{\mathbf{w}}(y_{1:L}|D, x_{1:t})$  compute.
2. Approximate the task-specific distribution  $Q(y_{1:L}|x_{1:t})$  using the learned models  $P_{\alpha}(D|x_{1:t})$  and  $P_{\mathbf{w}}(y_{1:L}|D, x_{1:t})$ . Provide an intuitive explanation for your derivation.

## 2 PROGRAMMING EXERCISES

**QUESTION 1 [25 Points] (Basic Context-Aware LM)** In this question, we implement a basic context-aware language model. For simplicity, we use character-level tokenization. Each token  $x_t$  is embedded by an embedding of size  $E$ . The context is then built by averaging all previous embeddings. Denoting the embedding at time  $i$  by  $\mathbf{e}_i$ , the context at time  $t$  is computed as

$$\mathbf{c}_t = \frac{1}{t} \sum_{i=1}^t \mathbf{e}_i.$$

The *logits* for the output probability distribution is then computed by a linear layer  $\mathbf{W} \in \mathbb{R}^{I \times E}$ , where  $I$  denotes the vocabulary size. This means that the final layer computes

$$\mathbf{z}_t = \mathbf{W}\mathbf{c}_t,$$

and passes it through a softmax  $(\cdot)$  to compute the distribution of next token.

1. Complete Question 1 in `Lastname_Firstname_Asgn1.ipynb` to implement this language model.
2. Answer all parts marked by `# COMPLETE`.

**QUESTION 2 [25 Points] (BERT Model)** In this question, we work with the pretrained BERT model [1]. BERT is a pretrained transformer-based language model, which uses bidirectional (i.e., without masked decoding) attention computations to compute context. We can however use it for text generation by using the special token `[MASK]` as illustrated in the assignment notebook. In this question, we first use the pretrained BERT to complete a text. We then *selectively* fine-tune BERT for text classification on a small dataset collected from IMDB dataset. For fine-tuning, we use the low rank adaptation (LoRA) method. Details of each part are given in the assignment notebook.

1. Complete Question 2 in `Lastname_Firstname_Asgn1.ipynb` to use BERT of text completion and classification.
2. Answer all parts marked by `# COMPLETE`.

## REFERENCES

- [1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proc. Conference of the Association for Computational Linguistics: Human Language Technologies (North American Chapter)*, pages 4171–4186, 2019.
- [2] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474, 2020.