Practice Exam Questions

Final Exam Date: Mon 05/04, 2026 7:00p - 9:00p @ PHYS 203

Below are a sample of practice exam questions from previous years. Note that the set of covered topics can vary from year to year, and so the below list may contain questions on topics that are not covered this year (and therefore, will not be tested). The length of this document is **not** indicative of the length of the exam.

Short Questions

Q: **Define.** What is a language model?

Q: "Teacher forcing" refers to the training approach where the model is trained with a loss function that only depends on the correct (i.e., ground truth) label. Why is teacher forcing used when pre-training a language model with the standard next-token language-modeling objective?

Q: Explain shortly what the n-gram assumption is.

Q: What is the purpose of the residual connections in the transformer?

Q: Give examples of two sentences that have high BLEU score but are semantically very different.

Q: **True or false:** LoRA is a method used to approximate changes to weight matrices in Transformers to lower the memory cost during pre-training.

Q: What is the purpose of the causal attention mask in autoregressive (decoder-only) language models?

Q: (4 points) Explain the benefits and drawbacks of training a transformer using an extremely large and extremely small batch size.

Q: Let's say we train a language model with $N = 10^8$ parameters and a training data set of size $D = 10^9$ for $T = 10^5$ iterations of gradient descent, and we obtain cross-entropy loss $L = 3.3$ on validation data. Let's say we want to scale up the model so that it achieves a lower cross-entropy loss of $L' = 0.7$. **True or false:** If we increase the model size N, data set size D, and training iterations N to be sufficiently large, we can obtain a model with cross-entropy loss $L' = 0.7$. Explain.

Q: Consider the following NLP methods: (1) bi-gram language model implemented using a feedforward neural network (assume a single hidden layer) (2) Word2Vec embedding. Identify one aspect in which the two methods are similar and one in which they are different.

Long Question 1

Suppose we implement an n-gram model using an MLP. The inputs are 1-hot encoded tokens, and so the input dimension of the MLP is nV where V is the size of the vocabulary. There is one hidden layer with dimension d, followed by the output layer. The MLP applies a ReLU on the activations in the hidden layer.

Q: What is the size of the output layer?

Q: How many parameters are in this model?

Q: How will the model behave if d is set too small?

Q: How will the model behave if d is set too large?

Q: How will the model behave if n is set too small?

Q: Suppose we have a pretrained model where n, V, and d are sufficiently large such that full fine-tuning is not feasible due to GPU memory constraints, but you have sufficient memory to load the model and run forward passes. Given a supervised fine-tuning dataset, how would you modify the model to enable you to fine-tune it? Be specific about which parameters you would add/modify/remove.

Long Question 2

Consider the following context-free grammar:

S → NP VP	V → 'ate'
VP → V	NN → 'John'
VP → V NP	NN → 'Mary'
VP → VP PP	NN → 'glasses'
NP → NN	NN → 'kitchen'
NP → NP PP	DT → 'the'
PP → IN NP	IN → 'with'
V → 'sees'	IN → 'in'

Q: Given the sentence 'John sees Mary with glasses', draw **two** valid parse trees according to the grammar above.

Q: Suppose we add the rules $NN \rightarrow$ 'We' and $V \rightarrow$ 'see' to the grammar. Give an example of a sentence in the language of the resulting grammar that is not grammatical with respect to English.

Q: How could you modify the grammar so that the ungrammatical sentences in the above question are excluded from the language? (without excluding grammatical sentences such as 'We see John with glasses')

Long Question 3

The textual entailment is a general language understanding problem, in the sense that many other semantic understanding tasks can be reduced to it. In textual entailment, we are given

a premise P and a hypothesis H. The task is to determine whether P entails/implies H, P contradicts H, or neither.

Let's think about the coreference resolution problem. Consider the example "The hawk couldn't catch up to the rabbit because it was too fast." All the bold words and all the underlined words are coreferent (i.e., mentions of the same entity).

Q: Can you rewrite this problem as a textual entailment task? You can use this example to explain your work.

Q: We have trained our textual entailment system over a large collection of examples, such as the SNLI dataset. The data contains examples such as

```
{'premise': 'The sisters are hugging goodbye while holding to-go
             packages after just eating lunch.'
 'hypothesis': 'Two women are embracing while holding to-go
                packages.'
 'label': entailment}
```

Since this is a large collection (>500K pairs) we were able to train a very strong entailment system, that comes close to human performance on this dataset. Does this mean that coreference resolution is a solved task? Explain.

Q: Consider the problem of resolving prepositional attachment ambiguity, such as in the sentences "Sally saw Alex with binoculars." Can we write a textual entailment example that would resolve this ambiguity? If so, how?

Q: Give an example of an NLP task that can not be easily reduced to textual entailment.

Long Question 4

You are building a medium-size transformer-based generative language model for checking the *truth*-value of mathematical statements. Each example contains a mathematical statement and a label indicating whether the statement is `true` or `false`. For example:

> Let $A \in \mathbb{R}^{n \times n}$. If A^2 is diagonalizable over $\mathbb{R}$, then is A diagonalizable over $\mathbb{R}$?

You are given:

- a pretrained language model as the target model, with access to its parameters for fine-tuning (e.g. `Phi-4`);

- train/dev/test data containing mathematical statements, each with a `true`/`false` label;
- API access to a powerful LLM (e.g. `GPT-5.5`) during training, but not during final inference;
- limited GPU memory for training and deployment.

Assume that you have tried directly fine-tuning the target model on the given labeled training data of mathematical statements, but the resulting model gives poor performance, because the problems require mathematical reasoning.

Q: (3 points) For a decoder-only Transformer model, what training objective is typically used during pre-training?

Q: (3 points) During training, you try to train the target model entirely using `bfloat16` precision to save memory. The training loss initially decreases, but later suddenly increases, even though the learning rate is small enough that the spike in the loss is not caused by taking steps that are too large. Conceptually, what can cause this instability? Why does a small learning rate not fully solve the issue?

Q: (3 points) Suggest one way to improve training stability without reducing the number of model parameters. Explain the basic idea and the memory tradeoff.

Q: (3 points) After training, you only need to run inference. Compare quantizing the model to `int8` versus `int4`/`float4`. Assume the GPU supports accelerated matrix operations for `int8`, but does **not** support accelerated matrix operations for `int4` or `float4`.

Q: (4 points) Since the math problems require reasoning, you decide to use the powerful LLM as a teacher during training. Describe a fine-tuning strategy for the target model. Your answer should specify:

- what you ask the powerful LLM to produce;
- what the target model receives as input;
- what the target model is trained to output;
- how you make sure the final `true`/`false` label can be extracted during inference.

Q: (4 points) API calls to the powerful LLM are expensive. Give one way to reduce the number of teacher calls while still improving the target model. Also explain how access to the teacher model's output probabilities/logits could make training more efficient.