Machine Learning Engineer

Take Home Exercise Overview

Sentence Transformers & Multi-Task Learning:

Objective:

The goal of this exercise is to assess your ability to implement, train, and optimize neural network architectures, particularly focusing on transformers and multi-task learning extensions. Please explain any and all choices made in the course of this assessment.

Task 1: Sentence Transformer Implementation

Implement a sentence transformer model using any deep learning framework of your choice. This model should be able to encode input sentences into fixed-length embeddings. Test your implementation with a few sample sentences and showcase the obtained embeddings. Describe any choices you had to make regarding the model architecture *outside of the transformer backbone.*

Task 2: Multi-Task Learning Expansion

Expand the sentence transformer to handle a multi-task learning setting.

1. **Task A:** Sentence Classification – Classify sentences into predefined classes (you can make these up).
2. **Task B:** [Choose another relevant NLP task such as Named Entity Recognition, Sentiment Analysis, etc.] (you can make the labels up)

Describe the changes made to the architecture to support multi-task learning.

Task 3: Training Considerations

Discuss the implications and advantages of each scenario and explain your rationale as to how the model should be trained given the following:

1. If the entire network should be frozen.
2. If only the transformer backbone should be frozen.
3. If only one of the task-specific heads (either for Task A or Task B) should be frozen.

Consider a scenario where transfer learning can be beneficial. Explain how you would approach the transfer learning process, including:

1. The choice of a pre-trained model.
2. The layers you would freeze/unfreeze.
3. The rationale behind these choices.

Task 4: Training Loop Implementation (BONUS)

If not already done, code the training loop for the Multi-Task Learning Expansion in Task 2. Explain any assumptions or decisions made paying special attention to how training within a MTL framework operates. Please note you need not actually train the model.

Things to focus on:

- Handling of hypothetical data
- Forward pass
- Metrics

Submission Instructions:

Share your code in a well-organized git repository, for extra points package it up in a Docker container. For each task, provide explanations or comments for the steps you've taken. This is essential for us to understand your thought process.

For Task 3 and Task 4, besides the technical explanation, also provide a brief write-up summarizing your key decisions and insights.

Ensure you include a requirements.txt file or an equivalent environment setup method so that we can replicate your results.

Evaluation Criteria

- Quality, depth, and clarity of explanations.
- Ability to make and justify design decisions.
- Clarity and structure of the code.
- Efficiency of the implemented models.
- Good luck! We look forward to reviewing your solutions.

AI Statement:

Note: We kindly ask that you refrain from relying on tools like ChatGPT, Copilot, or similar tools to complete this assessment for you. These are great tools for day-to-day velocity accelerations tasks (our engineers totally use them), but they can get in the way of seeing how you truly think about breaking down new technical challenges like the ones we solve every day at Fetch.

Additional Information:

We do not assign deadlines to this exercise because we understand it is a bit more of a time investment than a typical technical assessment or coding test. However, if you think you will need more 1-2 weeks to find the time to tackle this take home, please let us know.

We recommend giving yourself a 2 hour time limit to work on the assessment but you are not evaluated on how quickly you are able to turn around a project. Make sure to take time to test your work and provide quality written responses to answers.

We should be able to circle back within a week of your submission. Your recruiter will follow up if there are any unforeseen bottlenecks that may extend that timeline.