Programming Assignment 1

Query Likelihood Model and LMs

Atishay Jain (210050026), Chaitanya Garg (210050039)

14th September 2024

1 Methodology

1.1 Data Preparation

- **Dataset Overview:**
  - Training data contained 7 negative samples per positive instance.
  - Test data contained 15 negatives per positive.
  - Negative Sampling Methods:
    - Random: A corpus is created using all datapoints and negatives for each datapoint is sampled from this global corpus.
    - In-batch: Datapoints are divided into groups of `batch size`, and the corpus created within the group is used to sample negatives for datapoints in the group.
    - Batch size of 32 was chosen for in-batch negative generation.
  - In the test dataset:
    - Each data point had one query/document, a positive, and 7 negatives.
    - Multiple data points could be generated for the same query/document.
  - In the test data:
    - Each data point had one query/document, all corresponding positives, and 15 times that negatives.
- **Preprocessing:**
  - Each (query, document) pair, including both positives and negatives, was tokenized using BERT's tokenizer, and padded uniformly.
  - The datasets stored the tokenized pairs, and retrieving an item from the dataloader provided ready-to-use inputs for the model.

1.2 Model Architecture

- **Base Model:**
  - We used `BertForPreTraining` for the 1st and 3rd parts of the task, where similarity is calculated based on logits.
  - `BertModel` is employed for the 2nd and 4th parts, where a feed-forward neural network (FFN) is used to compute similarity.
- **Intermediate Layer Logits:** `BertModel` consists of 12 layers. For the extra credits part, we used the output of layer 6 instead of final output, and used FFN to get similarity.

- **Feedforward Layer (FFL) Configuration:** Our feedforward network with a depth of 1, with the following structure.
  - **Linear layer**: output the same size as the input.
  - **Batch Normalization**: to stabilize training and improve convergence.
  - **ReLU activation**: for non-linearity.
  - **Dropout layer**: to prevent overfitting.
  - **Final Linear layer**: outputs a single similarity score.

1.3 Training Procedure

- **Freezing BERT Layers:**
  - Out of the 12 layers, the first 9 layers are frozen.
  - Prevents overfitting and reduce computational complexity.
  - Leverages pre-trained knowledge from early layers, while fine-tuning the deeper layers for the specific task.
- **Loss Function**: Utilizes Cross Entropy Loss as specified in the problem statement to evaluate model performance.
- **Optimizer**: Employs the `AdamW` optimizer for training, which includes weight decay to improve generalization and optimize the model.
- **Batching Strategy**: Each datapoint, consisting of 1 positive and 7 negatives, is passed as a batch, with size 8.
- **Epochs**: The model is trained for 3 epochs to balance the stability of the loss and compute time.

1.4 Evaluation Procedure

- **Metrics Used**: Precision@1, Precision@10, MRR, MAP.
- For both Query Likelihood and Document Likelihood settings, each data point includes a **single query** along with **multiple positives and negatives**.
- **Batching Strategy**:
  - Given that the number of negatives per positive is 15, and a single data point might include numerous positives, treating a single data point as a batch leads to CUDA running out of memory.
  - Therefore, the data is split into batches of size 8 to manage memory usage effectively.

2 Experiments and Results

2.1 HotpotQA Dataset

	Precision@1	Precision@10	MRR	MAP
Query Likelihood Logits	0.9600	0.2393	0.9783	0.9343
Query Likelihood FFL	0.9812	0.2410	0.9823	0.9729
Document Likelihood Logits	0.9850	0.2409	0.9919	0.9675
Document Likelihood FFL	0.9975	0.2416	0.9984	0.9856

Table 1: Random sampling negatives from whole corpus

	Precision@1	Precision@10	MRR	MAP
Query Likelihood Logits	0.9082	0.2401	0.9514	0.9073
Query Likelihood FFL	0.9382	0.2302	0.9685	0.9203
Document Likelihood Logits	0.9167	0.2392	0.9533	0.9065
Document Likelihood FFL	0.9708	0.2414	0.9851	0.9591

Table 2: Inbatch sampling of Negatives

2.2 WikiNQ Dataset

	Precision@1	Precision@10	MRR	MAP
Query Likelihood Logits	0.1941	0.1782	0.3598	0.2896
Query Likelihood FFL	0.1953	0.1695	0.3624	0.2948

Table 3: Random sampling negatives from whole corpus

	Precision@1	Precision@10	MRR	MAP
Query Likelihood Logits	0.2388	0.1900	0.4015	0.3206
Query Likelihood FFL	0.2471	0.1789	0.4356	0.3623

Table 4: Inbatch sampling of Negatives

	Precision@1	Precision@10	MRR	MAP
Document Likelihood Logits	0.2433	0.1742	0.4032	0.3061
Document Likelihood FFL	0.2892	0.2232	0.4597	0.3890

Table 5: Negative samples already provided in the dataset

3 Observations

Added some points from me and GPT. Need to modify as needed

- The positive and negative examples in WikiNQ are very similar, with only small differences between them. This makes it harder for the model to capture the deeper context needed to distinguish between them, leading to worse results compared to HotpotQA.
- Global sampling performs better than in-batch sampling, likely because the negatives are drawn from a larger corpus, offering a wider variety of contexts and making it easier for the model to learn to differentiate between relevant and irrelevant documents.
- FFL-based scoring outperforms logits-based scoring, possibly because FFL can learn a more complex model that identifies which aspects of the model output are most relevant for determining the similarity between the document and query, rather than relying on a fixed, logits-based approach.
- The addition of a feedforward layer (FFL) improved the model's capacity to capture complex relationships in the [CLS] token representation from BERT. While a single feedforward layer boosted the model's performance, increasing the depth beyond 2 layers resulted in minimal gains and sometimes even caused a slight increase in loss due to overfitting.

- Freezing some of the early layers of the BERT model had a noticeable impact on training time and performance. By freezing the first 9 layers, the training process became faster without significantly affecting performance, indicating that these early layers primarily capture general language features that don't need fine-tuning for this task. This freezing reduced the training time by approximately 40% without any notable drop in model performance, suggesting that fine-tuning only the last few layers of BERT is sufficient for this document similarity task.
- One of the challenges faced was the fluctuation in loss after the first epoch. In some cases, the loss increased significantly due to the model overfitting on the training data. This was mitigated by adding dropout, freezing layers, and using a lower learning rate.

4 Extra Credit

For the extra credit part, we ran the FFL parts by extracting BERT's intermediate output from 6th layer instead of the last layer

	Dataset and -ve sampling	Precision@1	Precision@10	MRR	MAP
Query Likelihood FFL	HotpotQA-Random	0.9273	0.1288	0.9537	0.9522
	HotpotQA-Inbatch	0.9245	0.1306	0.9525	0.9512
	WikiNQ-Random	0.7154	0.1503	0.8037	0.7993
	WikiNQ-Inbatch	0.7227	0.1571	0.8158	0.8150
Document Likelihood FFL	HotpotQA-Random	0.8142	0.2202	0.8800	0.7667
	HotpotQA-Inbatch	0.7483	0.2189	0.8420	0.7286
	WikitNQ	0.2417	0.1842	0.4035	0.3231

Table 6: Taking BERT output from intermediate layer 6

- We also tried to do it for different layers like 7, 8, 9 and train on a smaller dataset to see results (as training takes a lot of time over whole data). We saw that as me increased the layer from which output is taken from 6 to 9, the performance was improving
- The losses showed slower convergence when using outputs from the 6th layer, suggesting that more training data and additional epochs might be required for better performance. In contrast, using the 12th layer provided more refined features, leading to faster convergence with fewer epochs.