NLP and Information Retrieval
Sample Exam 2 with Answers

Generated by ChatGPT

Instructions

- Answer all questions.
- Show all calculations where required.
- Justify your answers clearly in transfer questions.

Task 1: Fundamentals of NLP

1a) (Knowledge)
Define the following terms in one or two sentences each:

- **Token**
- **Type**
- **Vocabulary**

Answer:
A **token** is a single instance of a sequence of characters (such as a word) extracted from text. A **type** is a unique token, and the **vocabulary** is the set of all distinct types in a corpus.

1b) (Understanding)
Why might **linguistic preprocessing** (e.g., lowercasing, stopword removal) sometimes **harm** performance in deep learning pipelines, even though it helps in classical IR?

Answer:
Excessive preprocessing can remove valuable information (such as capitalization cues, nuances in stopwords, or morphological details) that deep learning models rely on to learn rich contextual representations.

Task 2: Comparison of IR Models

2a) (Knowledge)
Contrast the **vector space model** (TF-IDF) and the **probabilistic model** (BM25). How does BM25 address some shortcomings of a raw TF-IDF approach?

Answer:
The vector space model (using TF-IDF) represents documents as weighted vectors and uses cosine similarity, whereas BM25 incorporates term frequency saturation, document length normalization, and refined inverse document frequency to better capture the relevance of terms across documents.

2b) (Transfer)
You run a small search engine for **multilingual** documents. Would you rely solely on BM25, or incorporate a semantic embedding-based technique? Justify your decision.

Answer:
While BM25 is fast and effective for lexical matching, multilingual documents often require semantic understanding to capture cross-language synonyms; thus, incorporating semantic embeddings (e.g., via dense retrieval) can improve retrieval performance despite higher resource requirements.

Task 3: True/False – Transformer & RNN

Mark each statement **True (T)** or **False (F)** and provide one sentence of explanation:

(1) RNN-based LMs cannot model long-range dependencies at all.

(2) Transformers require less memory than RNNs for the same sequence length.

(3) Self-attention attends to all positions in the sequence simultaneously.

(4) A GRU has more parameters than an LSTM for the same hidden size.

(5) Transformers rely on positional encodings to track word order.

Answers:

(1) **False.** RNNs can model long-range dependencies but often suffer from vanishing gradients, making them less effective than Transformers.

(2) **False.** Transformers typically require more memory due to the quadratic complexity of self-attention with respect to sequence length.

(3) **True.** Self-attention computes interactions between all pairs of tokens in the sequence simultaneously.

(4) **False.** GRUs are generally designed with fewer parameters than LSTMs because they use a simpler gating mechanism.

(5) **True.** Since self-attention is permutation-invariant, Transformers add positional encodings to maintain the order of tokens.

Task 4: N-Gram Smoothing

4a) (Knowledge)
What problem does **smoothing** solve in an n-gram language model, and why is it crucial for robust probability estimates?

Answer:
Smoothing addresses the zero-probability problem for unseen n-grams by redistributing some probability mass to them, ensuring that the model can assign non-zero likelihoods even for novel sequences.

4b) (Short Calculation)
Given a **bigram LM**, you have the counts:

- $C(\text{the, cat}) = 10$
- $C(\text{the, dog}) = 0$
- $C(\text{the}) = 25$

Apply **add-1 (Laplace) smoothing** to estimate $P(\text{dog} \mid \text{the})$. Show your steps.

Answer:
Using add-1 smoothing:

$$P(\text{dog} \mid \text{the}) = \frac{0+1}{25+V},$$

where V is the vocabulary size. For instance, if $V = 1000$, then $P(\text{dog} \mid \text{the}) = \frac{1}{1025}$.

Task 5: Byte-Pair Encoding (BPE)

5a) (Understanding)
Explain how **Byte-Pair Encoding** (BPE) is built step-by-step. Why does it help reduce **out-of-vocabulary** issues?

Answer:
BPE starts with a base vocabulary of individual characters and iteratively merges the most frequent adjacent symbol pairs into new tokens; this process allows rare words to be represented as a sequence of common subword units, reducing OOV occurrences.

5b) (Mini Exercise)
Given the symbols $\{l, o, v, e, \_\}$ (where $\_$ represents a space), process the string "love love l ove" with one or two hypothetical merge steps. Show the updated tokenization.

Answer:
A possible first merge could combine "l" and "o" into "lo," transforming the string to "lo ve lo ve l ove." A subsequent merge could combine "lo" and "ve" into "love," resulting in tokens like "love love l ove," thereby reducing the number of tokens required.

Task 6: Retrieval Evaluation – MRR & nDCG

6a) (Calculation)
A ranked list of 5 documents has the following relevance:

- D1: relevant
- D2: relevant
- D3: not relevant
- D4: relevant
- D5: not relevant

Compute the **Mean Reciprocal Rank (MRR)**.

Answer:
MRR is calculated using the reciprocal rank of the first relevant document. Here, the first relevant document is at rank 1, so MRR = 1.0.

6b) (Understanding)
How does **nDCG** differ from MRR? When is nDCG preferable?

Answer:
nDCG accounts for the graded relevance and positions of all relevant documents, making it suitable when multiple relevant documents exist and their order matters, unlike MRR which only considers the first relevant document.

Task 7: Dense Retrieval & Approximate Nearest Neighbor

7a) (Knowledge)
Describe briefly how **ANN structures** like HNSW or Faiss speed up similarity searches in high-dimensional embeddings.

Answer:
ANN structures organize data into graphs or clusters that allow the search algorithm to quickly narrow down the candidate set, drastically reducing the number of comparisons needed to find the nearest neighbors.

7b) (Transfer)
You plan to handle **100 million** paragraphs in a news archive. Which **ANN structure** (e.g., HNSW, IVF in Faiss, etc.) would you pick and why?

Answer:
For such a large dataset, an IVF (inverted file) index in Faiss is often preferred because it scales well by partitioning the dataset into clusters, balancing memory usage and search speed, though HNSW might be chosen if higher accuracy is needed.

Task 8: Neural Re-Ranking with BERT

8a) (Understanding)
In a **two-stage retrieval** pipeline, how are query and document tokens typically combined as input for BERT?

Answer:
The query and document tokens are concatenated into a single sequence, separated by special tokens (e.g., starting with [CLS], then the query, followed by [SEP], and finally the document), allowing BERT to model their interactions.

8b) (Transfer)
Your e-commerce store has high user traffic. If you add a BERT re-ranker to rank the top 100 results, latency becomes a problem. Suggest **two optimizations**.

Answer:
One optimization is to reduce the number of candidates passed to BERT by using a stricter first-stage filter. Another is to apply model distillation or quantization to speed up inference while maintaining acceptable accuracy.

Task 9: Instruction Tuning

9a) (Knowledge)
What is **instruction tuning**? How does it differ from using long prompts with an unmodified model?

Answer:
Instruction tuning involves fine-tuning a model on a dataset of instruction-response pairs so that its parameters adapt to better follow instructions, as opposed to using long prompts which do not alter the model's internal weights.

9b) (Transfer)
Your **FAQ chatbot** must follow formatting instructions accurately. Describe **two ways** instruction tuning can improve consistency, and mention **one limitation**.

Answer:
Instruction tuning can help by training the model to consistently produce outputs in the desired format and to handle diverse instruction types effectively. However, a limitation is that collecting a high-quality, diverse instruction-response dataset can be resource-intensive.

Task 10: RLHF & Model Alignment

10a) (Understanding)
In **RLHF**, what is the role of the **reward model**, and why use **pairwise** human feedback?

Answer:
The reward model estimates how well an output aligns with human preferences, and pairwise feedback is used because it is generally easier and more reliable for humans to compare two outputs rather than assigning absolute scores.

10b) (Transfer)
Your chatbot produces **toxic** outputs in rare cases. How would you build a **human preference dataset** to mitigate this behavior? Mention one risk.

Answer:
I would collect data where human annotators compare pairs of outputs to indicate which is less toxic, then use this data to train the reward model; a risk is that annotator bias might skew the model's perceptions of what constitutes toxicity.

Task 11: Handling Long Context in Transformers

11a) (Knowledge)
What problem arises in **self-attention** when handling long sequences?

Answer:
Self-attention scales quadratically with sequence length, resulting in high computational and memory demands for very long inputs.

11b) (Transfer)
You need to answer questions about **25,000-token transcripts**. Would you use **sparse-attention Transformers** or **retrieval-augmented generation (RAG)**? Justify.

Answer:
I would choose RAG, as it allows the system to retrieve the most relevant segments from long transcripts, thereby reducing memory load and computational cost while maintaining high accuracy; sparse-attention Transformers can process long sequences but may still be less efficient in practical scenarios with dynamic document updates.