NLP and Information Retrieval
Sample Exam 3 with Answers

Generated by ChatGPT

Instructions

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

Task 1: Terminology & Basic Concepts

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

- **Precision**
- **Recall**
- **F1 Score**

Answer:
Precision is the fraction of retrieved documents that are relevant. **Recall** is the fraction of all relevant documents that are retrieved. **F1 Score** is the harmonic mean of precision and recall, balancing the trade-off between the two.

1b) (Understanding)
Why might **F1** be more informative than **Accuracy** in scenarios where the dataset is highly **imbalanced**?

Answer:
In imbalanced datasets, high accuracy can be misleading because a model might simply predict the majority class; F1 score better reflects the performance on the minority class by considering both precision and recall.

Task 2: True/False – Foundations

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

(1) Lemmatization always strips affixes down to the root form, regardless of context.

(2) In a pipeline for text classification, feature extraction is optional if using a bag-of-words approach.

(3) Boolean Retrieval evaluates partial matches by computing similarity scores for each document.

(4) Kneser-Ney smoothing is a method used in language modeling to handle unseen n-grams.

(5) A GloVe embedding can dynamically change depending on the words around it in a sentence.

Answers:

(1) **False.** Lemmatization uses context to return the proper base form rather than simply stripping affixes.

(2) **False.** Even with a bag-of-words approach, feature extraction (e.g., tokenization, vectorization) is essential to convert text into numerical representations.

(3) **False.** Boolean retrieval uses binary matching (presence/absence) without computing graded similarity scores.

(4) **True.** Kneser-Ney smoothing redistributes probability mass to unseen n-grams, ensuring that they receive non-zero probabilities.

(5) **False.** GloVe embeddings are static; they do not change with context, unlike contextual embeddings such as those from BERT.

Task 3: Naïve Bayes & Text Classification

3a) (Knowledge)
Briefly outline how a **Naïve Bayes classifier** estimates $P(\text{Class} \mid \text{Features})$. Why is the assumption called “naïve”?

Answer:
A Naïve Bayes classifier applies Bayes’ theorem under the assumption that features are conditionally independent given the class, an assumption considered ”naïve” because in practice features are often correlated.

3b) (Transfer)
You are performing **sentiment analysis** (positive vs. negative) on product reviews. Given a small training set, would you choose **Naïve Bayes** or a **Transformer-based** classifier? Justify your choice regarding data size, complexity, and potential accuracy.

Answer:
With a small dataset, Naïve Bayes is preferred because it is simple, computationally efficient, and less prone to overfitting, whereas Transformer-based models require large amounts of data and computational resources to perform well.

Task 4: Inverted Index & IR

4a) (Knowledge)
Explain how an **inverted index** is constructed. Include the terms: posting list, document frequency (DF), and term frequency (TF).

Answer:
An inverted index is built by processing a collection of documents and mapping each unique term to a **posting list** that contains the IDs of documents in which the term appears, along with the term frequency (TF) in each document; the document frequency (DF) is the number of documents in which the term appears.

4b) (Mini Exercise)
You have the following small corpus:

- **D1**: "Time flies like an arrow"
- **D2**: "Fruit flies like a banana"

Lemmatize and ignore function words ("a", "an"). Construct the **inverted index** for the terms "time", "fly", "like", "arrow", "fruit", "banana". Show the postings lists.

Answer:
After lemmatization and removal of function words, assume D1 becomes {"time", "fly", "like", "arrow"} and D2 becomes {"fruit", "fly", "like", "banana"}. The inverted index is:

- **time**: {D1}
- **fly**: {D1, D2}
- **like**: {D1, D2}
- **arrow**: {D1}
- **fruit**: {D2}
- **banana**: {D2}

Task 5: BM25 Calculation (Short Exercise)

You have a collection of three documents (D1, D2, D3). Each term's frequency and document lengths are as follows:

- **D1**: length = 4 tokens; term "cat" appears **2** times.
- **D2**: length = 8 tokens; term "cat" appears **1** time.
- **D3**: length = 8 tokens; term "cat" appears **0** times.

Assume average document length (avgDL) = 6. The query is "cat", and we use BM25 with $k_1 = 1.2$ and $b = 0.75$.
Which document likely gets a **higher score** under BM25, D1 or D2? Provide a short reasoning without full numeric calculation.

Answer:
Document D1 is likely to score higher because its higher term frequency relative to its shorter length results in a stronger signal in BM25's scoring, compared to D2.

Task 6: Neural IR – Re-Ranking

6a) (Understanding)
In a **two-stage retrieval** pipeline with a **BERT re-ranker**, how are query and document tokens typically combined as input for BERT?

Answer:
They are concatenated into a single sequence with special tokens (e.g., starting with [CLS], then the query, followed by [SEP], and finally the document) so that BERT can capture interactions between the query and the document.

6b) (Transfer)
You have an **e-discovery** system for legal documents where high recall is mandatory. Explain why a BM25 first stage plus a BERT re-ranker might outperform a single-stage approach.

Answer:
The BM25 stage quickly filters a large collection to a manageable candidate set, and the BERT re-ranker then applies deep semantic understanding to accurately rank these candidates, combining speed with high-quality, context-sensitive retrieval.

Task 7: Dense Retrieval & Vectors

7a) (Understanding)
What does **"dense retrieval"** mean, and how does it differ from **sparse** approaches like BM25?

Answer:
Dense retrieval uses continuous vector representations (embeddings) to capture semantic similarity between queries and documents, unlike sparse methods like BM25, which rely on exact term matching and frequency counts.

7b) (Short Calculation)
Given a query embedding $\mathbf{q} = [0.6, 1.2]$ and two passage embeddings:

- $\mathbf{p}_1 = [1.0, 0.5]$
- $\mathbf{p}_2 = [0.0, 1.1]$

Compute the dot product for each passage. Which passage ranks higher?

Answer:

$$\mathbf{q} \cdot \mathbf{p}_1 = 0.6 \times 1.0 + 1.2 \times 0.5 = 0.6 + 0.6 = 1.2$$
$$\mathbf{q} \cdot \mathbf{p}_2 = 0.6 \times 0.0 + 1.2 \times 1.1 = 0 + 1.32 = 1.32$$

Since $1.32 > 1.2$, the passage corresponding to $\mathbf{p}_2$ ranks higher.

Task 8: Byte-Pair Encoding (BPE) & Subword Splits

8a) (Knowledge)
Why does **BPE** often outperform simple character-based tokenization in languages with rich morphology?

Answer:
BPE creates subword units that capture meaningful morphemes and patterns, enabling the model to represent rare or compound words more effectively than character-level tokenization, which lacks semantic structure.

8b) (Do Stuff)
You have the initial subword vocabulary $\{\_, a, t, o, r\}$ (where $\_$ represents a space). The text is:

"a t a t o r a tor"

Show one or two hypothetical merge steps and how they reduce the total token count.

Answer:
A possible first merge is combining "a" and "t" to form "at", so the text becomes: `"at at o r a tor"`. Next, merging "o" and "r" to form "or" changes it to: `"at at or a tor"`. These merges reduce the overall number of tokens by grouping frequent subword units.

Task 9: Instruction Tuning vs. Prompting

9a) (Understanding)
Explain how **instruction tuning** changes a model's behavior differently than simply providing a long prompt.

Answer:
Instruction tuning fine-tunes the model on a dataset of instruction-response pairs, thereby modifying its parameters to better adhere to given instructions; long prompts, on the other hand, only provide temporary context without altering the model's underlying behavior.

9b) (Transfer)
Your **FAQ chatbot** must handle a variety of user instructions. Outline two ways instruction tuning can help produce better, more consistent responses, and mention one limitation.

Answer:
Instruction tuning can standardize output formatting and improve the model's ability to follow diverse instructions, resulting in more consistent responses. However, a limitation is that it requires a large, high-quality instruction-response dataset, which can be expensive and time-consuming to collect.

Task 10: RLHF & Model Alignment

10a) (Understanding)
In **RLHF**, what is the role of the **reward model**, and why is pairwise human feedback often preferred over direct scoring?

Answer:
The reward model estimates the quality of a model's output based on human preferences and guides the reinforcement learning process. Pairwise feedback is preferred because it is generally easier and more consistent for humans to compare two outputs than to provide absolute scores.

10b) (Transfer)
Your chatbot occasionally produces **toxic** outputs. Describe how you would build a human preference dataset to mitigate this issue, and mention one potential risk.

Answer:
I would collect output pairs where human annotators select the less toxic version and use these comparisons to train the reward model, encouraging the chatbot to avoid toxic language. A potential risk is that annotator bias may skew what is considered toxic, potentially limiting the model's generalizability.

Task 11: Handling Long Context in Transformers

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

Answer:
Self-attention has quadratic time and memory complexity with respect to sequence length, which makes it computationally expensive and impractical for very long sequences.

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

Answer:
I would choose retrieval-augmented generation (RAG) because it allows the system to retrieve the most relevant segments from the transcript and focus on them, reducing memory usage and computational cost while maintaining high accuracy, compared to the still resource-intensive sparse-attention models.