Studying Sparse-Dense Retrieval: Part I

We will see examples of Maximum Inner Product Search (MIPS) over dense vectors and, separately, similar algorithms for sparse vectors. But what do we do when vectors have both a dense subspace and a sparse subspace? In other words, if your documents and queries are of the form $u \oplus v$, where $\oplus$ denotes concatenation, $u \in R^n$ is a dense vector, and $v \in R^N$ is a sparse vector, how do we efficiently retrieve the top-k documents with respect to a given query by maximal inner product?

One idea is to take advantage of the linearity of the inner product operator: The inner product of a query vector $q = q_{dense} \oplus q_{sparse}$ with a document vector $d = d_{dense} \oplus d_{sparse}$ is the sum of $q_{dense} \cdot d_{dense} + q_{sparse} \cdot d_{sparse}$. We can now approximate the joint MIPS by breaking it up into two smaller MIPS problems: Retrieve the top-k' documents from a dense retrieval system defined over the dense portion of the vectors, and another set of top-k' documents from a sparse retrieval system operating on the sparse portion, before somehow "merging" the two sets and retrieving the top-k documents from the combined (much smaller) set. As k' approaches infinity, the final top-k set will become exact, but retrieval becomes slower.

Your task is to examine the correctness of the algorithm above and study the effect of k' on the quality of the final top-k set, theoretically and/or empirically. If you wish to do a theoretical analysis, make sure your probabilistic model is clearly defined (i.e., state the distribution that governs the dense and sparse vectors) and you include experiments on synthetic data to corroborate your findings. If you wish to explore this empirically, design an experiment to demonstrate this effect on a real-world dataset with off-the-shelf deep learning "embedding" models.

Datasets

For experiments, you may use the MS MARCO Passage dataset which has about 8.8 million documents, or one of the BeIR datasets (recommended) which have a more manageable size and are easy to troubleshoot.

Models

To generate sparse vectors you are encouraged to encode every document using BM25, which can be expressed trivially as the inner product of two vectors (but include the proof in your report). To generate dense vectors, you are encouraged to use a lightweight model such as all-minilm-l6-v2 from the Sentence Transformers library.

Software

You're encouraged to use Python for this assignment. Working with Transformer-based models is relatively straightforward in Python, especially if you are using a pre-trained model from HuggingFace.

Metrics

You may measure the quality of a retrieved set in two ways:

- Ranking quality given relevance judgments: Both the BeIR and MS MARCO datasets have relevance judgments. You can thus use the official metrics such as MRR@k or NDCG@k to assess the quality of a retrieved set.
- Recall w.r.t. exact solution: Additionally, for every query, you can count the number of documents from the exact MIPS solution that appear in the approximate solution, and divide that by *k*. In other words, you're measuring whether the approximate algorithm "recalls" all the documents from the exact solution.

Format

You are expected to produce a technical report written in the style of an academic paper, with a detailed description of the problem, your approach, experimental setup, theoretical results and/or empirical observations.

Studying Sparse-Dense Retrieval: Part II

Suppose you have implemented the algorithm above to retrieve the top-*k* set by maximal inner product. Your task in this problem is to demonstrate that this sparse-dense retrieval can be beneficial in practice.

In particular, you are asked to study the effect of combining dense and sparse embedding models on the ranking quality on the MS MARCO or BeIR datasets. More concretely:

- Implement a program to perform retrieval over dense embeddings using FAISS;
- Implement another program (e.g., using WAND) to perform top-k retrieval over sparse vectors produced by BM2;
- Measure the quality of the two rankings separately using MRR@*k* or NDCG@*k* for some *k*;
- Examine if a fusion of the BM25 scores and dense model's scores lead to better ranking quality.

You are encouraged to think about the right *fusion*. As candidates, take a convex combination of BM25 and dense scores, $\alpha f_{Dense} + (1 - \alpha) f_{BM25}$, and study the effect of the parameter α on the final ranking quality. Does the fact that BM25 scores (or the dense scores) are unbounded pose an issue for the fusion? If so, how do you propose to remedy that?

Format

You are expected to produce a technical report written in the style of an academic paper, with a detailed description of the problem, your approach, theoretical results and empirical observations.