Neural IR

INFORMATION RETRIEVAL AND RECOMMENDER SYSTEMS

Georgios Peikos

University of Milano-Bicocca, Milan, Italy

Department of Informatics, Systems, and Communication (DISCo)

Today's Lecture – Neural IR Models

Why do we need Neural IR models?

What is a Neural IR model?

Types of Neural IR models

Neural IR in Search Systems

Efficiency and Deployment Considerations

Open Challenges and Further Reading

Training Neural IR models --- Can provide material if interested!

Why Do We Need Neural IR Models?

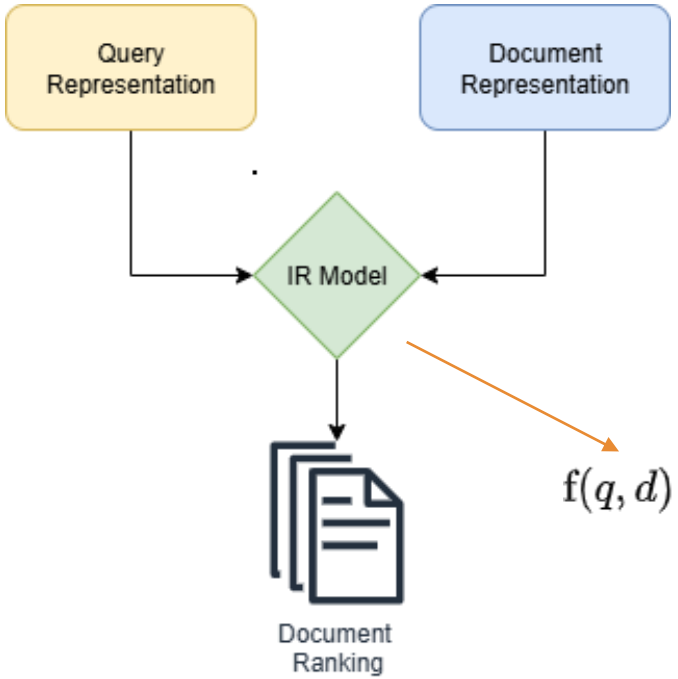
Information Retrieval Models

The goal of an IR system is to employ the **ranking function f** to generate a score f(q, d) for any query-document pair (q, d), reflecting **the (topical) relevance degree** between them, and produce a **relevance permutation πf(q, D)** according to the predicted score:

$$f(q, d) = g(\psi(q), \phi(d), \eta(q, d))$$

where **ψ**, **ϕ**, and **η** return **representations** of q, d, or **both**.

each of several possible ways in which
a set or number of things can be ordered



$$f(q, d) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i, d) \cdot (k_1 + 1)}{f(q_i, d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\mathrm{avgdl}}\right)}$$

Source: Robust Information Retrieval SIGIR 2024 tutorial

What is the problem with BM25?

From Lexical Matching to Neural Understanding

Models like **BM25** rely on **exact term matching**.

They fail when **query and document use different words** to refer to the same context (Vocabulary mismatch problem).

They cannot handle **polysemy** (same word, multiple meanings).

Example: “python” (language vs. animal).

They ignore the **order and structure** of words.

Query: "bank robbery suspect“

Document A: "suspect robbed a bank" (relevant)

Document B: "bank suspected of robbery" (less relevant)

Both might get similar scores despite different meanings.

From Lexical Matching to Neural Understanding

Context is ignored because each term is treated independently!

Semantic understanding, goes beyond mapping query terms to document terms.

Neural Information Retrieval leverages (deep) neural networks to encode queries and documents into a *shared semantic space*, enabling context-aware semantic retrieval.

! Neural IR models still estimate Topical Relevance!

What is a Neural IR Model?

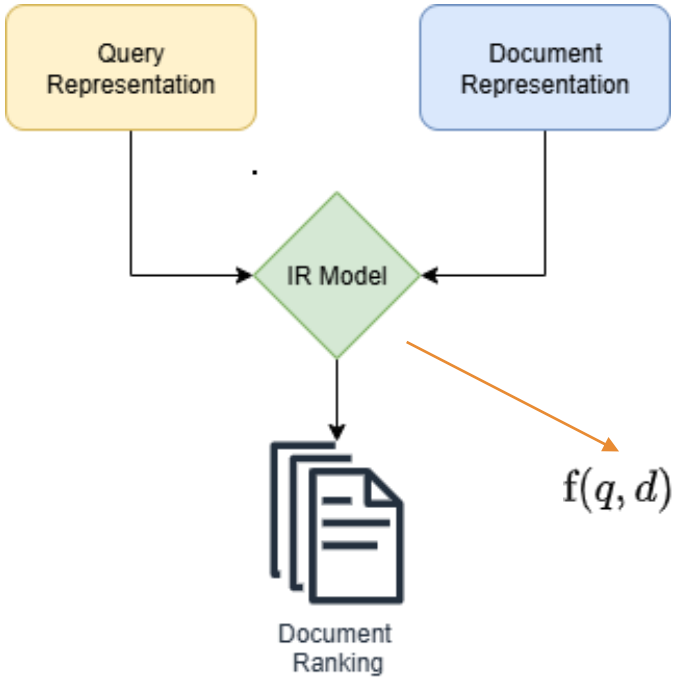
Information Retrieval Models

The goal of an IR system is to employ the **ranking function f** to generate a score f(q, d) for any query-document pair (q, d), reflecting **the (topical) relevance degree** between them, and produce a **relevance permutation πf(q, D)** according to the predicted score:

$$f(q, d) = g(\psi(q), \phi(d), \eta(q, d))$$

each of several possible ways in which
a set or number of things can be ordered

where **ψ**, **ϕ**, and **η** return **representations** of q, d, or **both**.



$$f(q, d) = \sum_{i=1}^{n} \mathrm{IDF}(q_i) \cdot \frac{f(q_i, d) \cdot (k_1 + 1)}{f(q_i, d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\mathrm{avgdl}}\right)}$$

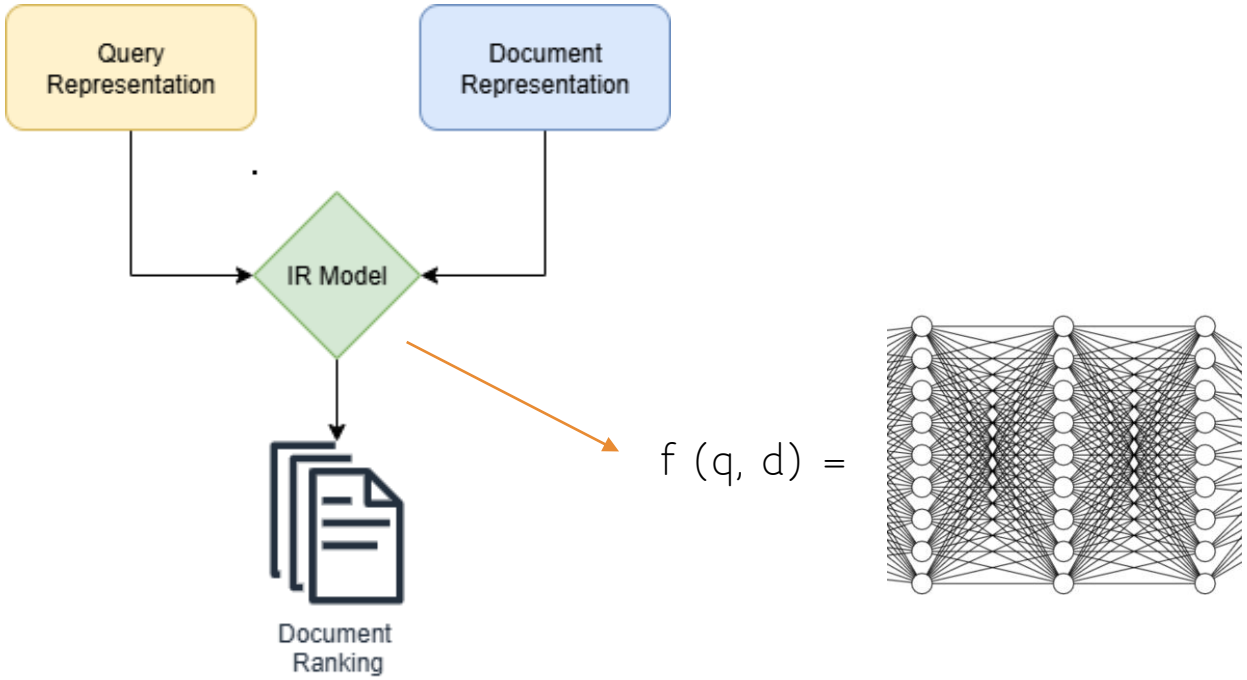
Neural Information Retrieval Models

The goal of an IR system is to employ the **ranking function f** to generate a score f(q, d) for any query-document pair (q, d), reflecting **the (topical) relevance degree** between them, and produce a **relevance permutation πf(q, D)** according to the predicted score:

$$f(q, d) = g(\psi(q), \phi(d), \eta(q, d))$$

each of several possible ways in which a set or number of things can be ordered

where **ψ**, **ϕ**, and **η** return **representations** of q, d, or **both**.



Neural Information Retrieval Models - Core Ideas

Neural IR models represent queries and documents into a continuous vector space.

Both queries and documents are represented using embeddings.*

The $g(\psi(q), \phi(d), \eta(q, d))$ functions are usually neural encoders (e.g., BERT).

However, these encoders should be **fine-tuned** for retrieval.

Fine-tuning for neural encoders for retrieval means teaching a neural model (by updating its weights) so that the embedding representations it creates for queries and documents **reflect topical relevance relationships.** **

In other words, after fine-tuning, the model learns to produce embeddings such that:

Relevant query-document pairs are close together in the **semantic** (vector) **space**.

Irrelevant pairs are far apart.

* Why embedding representation of text is better than bag of words representation? **How we fine-tune? (too be discussed)

Neural Information Retrieval Models - Core Ideas

If $f(q, d) = g(\psi(q), \phi(d))$, the neural IR model represents (encodes) the query $\psi(q)$ and the document $\phi(d)$ independently.

Here we have **one embedding** that represents the whole (all terms) **query**, and **one embedding** that represents the **whole document**.

The relevance score is obtained (usually) by estimating the cosine similarity of these two embeddings.

Neural Information Retrieval Models - Core Ideas

If $f(q, d) = g(\eta(q, d))$, the neural IR model represents (encodes) both **the query and the document content** into a single embedding.

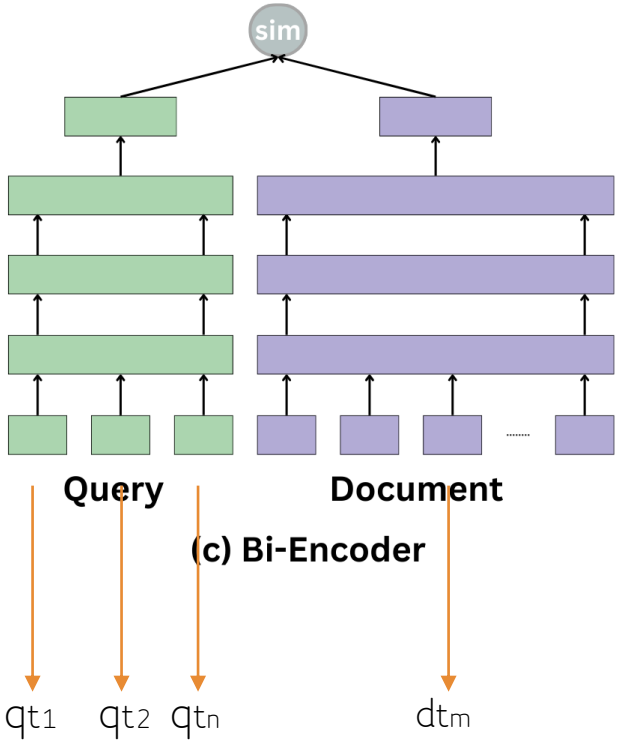
The relevance score is obtained from a specific layer e.g., from the final [CLS] output embedding.

Types of Neural IR models

Neural IR Models – Bi-Encoders

A Bi-Encoder is a **type** of neural IR model that encodes the query and the document independently into dense vector representations (embeddings), allowing the model to measure their **(topical) relevance** through vector similarity in a shared semantic space.

$$sim = f(q, d) = g(\psi(q), \phi(d))$$



Retrieval is fast because document embeddings can be precomputed (**offline**) and stored in a **vector index (also called** vector database**)**.

Similarity is computed using vector similarity e.g., **cosine similarity/dot product**.

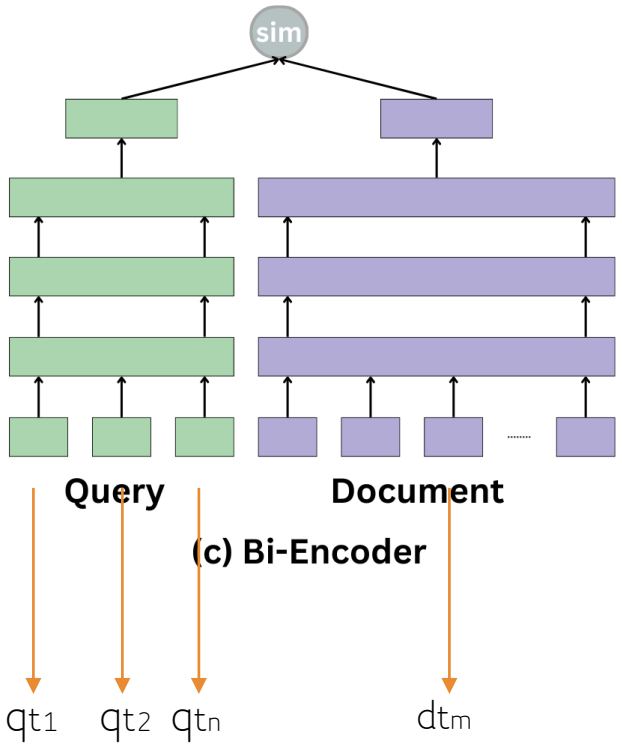
Suitable for **large-scale retrieval** (millions of documents).

A **single vector** must capture all relevant aspects of a long document, **which may lead to information loss**. For instance, BERT supports 512; Nowadays, 4,096 sequence length (tokens).

Relevance estimation may **not be very accurate**, as these architectures cannot **capture fine-grained token-level interactions** of **query** and **documents**!

Bi-Encoders – Models

$$sim = f(q, d) = g(\psi(q), \phi(d))$$



Sentence-BERT (SBERT)

Hofstätter, S., Lin, S. C., Yang, J. H., Lin, J., & Hanbury, A. (2021, July). Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval* (pp. 113-122).

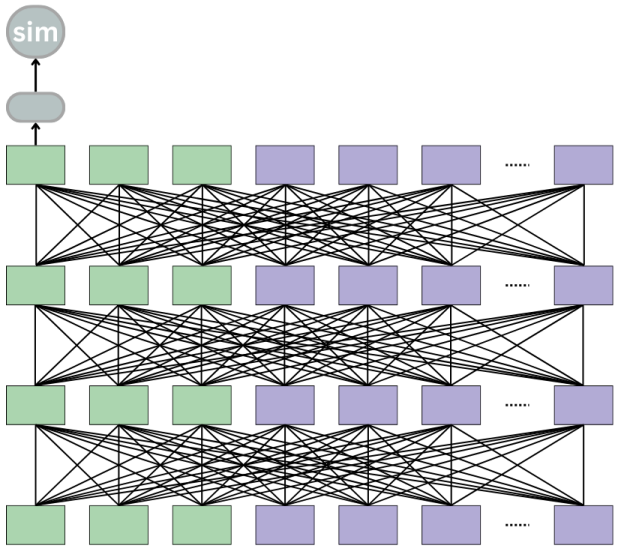
Contriever

Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., Grave, E.: Unsupervised dense information retrieval with contrastive learning. Transactions on Machine Learning Research (2022)

Neural IR Models – Cross Encoders

A Cross-Encoder is a type of neural IR model that **jointly** encodes the query and the document in the same input sequence, allowing the model to directly **capture rich, fine-grained interactions** between their tokens.

$$sim = f(q, d) = g(\eta(q, d))$$



(a) Cross Encoder

Such model **captures token-level interactions** between the query and the document. Typically outperforms bi-encoders in ranking quality.

The model learns to estimate relevance (during fine-tuning) directly rather than relying on vector similarity.

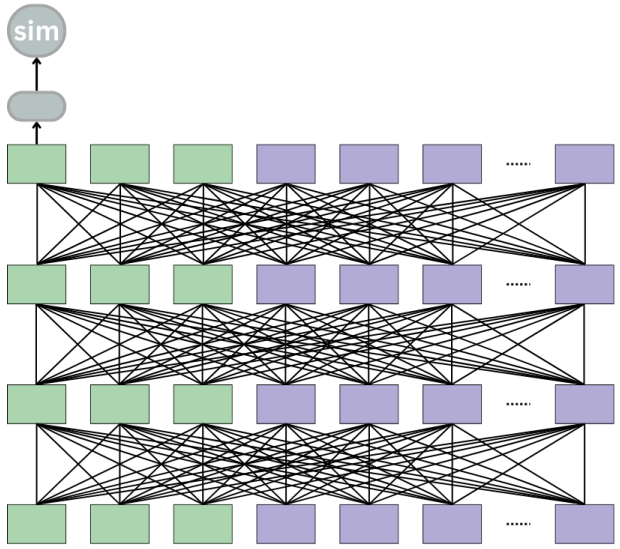
The query-document pair must be encoded together, meaning the model must process each candidate document separately (online!).

This makes cross-encoders impractical for large-scale retrieval, since it would require encoding millions of documents per query.

Suitable for re-ranking only a small set of candidate documents (e.g., top 100-1000 from a bi-encoder retriever or a probabilistic model).

Cross Encoders – Existing Models

$$sim = f(q, d) = g(\eta(q, d))$$



(a) Cross Encoder

MiniLM

Wang, W., Wei, F., Dong, L., Bao, H., Yang, N., & Zhou, M. (2020). Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, *33*, 5776-5788.

MonoT5

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In Findings of EMNLP, pages 708–718.

Efficiency and Deployment Considerations

Usage of Neural IR in Search Systems

Document Representation via indexing

Document Representation via a neural language model (embeddings)

Offline Process

Online Process

Query

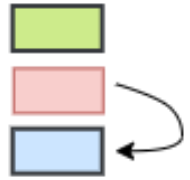
Query Representation using the index

Query Representation via the same neural model

1st Stage Retrieval

Ranked List

2nd Stage Retrieval



Ranked List

BM25 relies on Index

Bi-encoder relies on stored embedding representation

Expensive Cross-Encoder relies on original document Texts!

Usage of Neural IR in Search Systems

Storage: Neural retrievers require more space to **store dense embeddings.** Bi-encoder relies on an **Approximate Nearest Neighbor (ANN)** index for fast retrieval.

Latency: First-stage retrievers (BM25 or bi-encoder) must be optimized for low latency.

Cost: Cross-encoders increase inference time; hence, they are usually applied only on a small set of **candidate** documents.

Scalability: Approximate Nearest Neighbor indexes (e.g., FAISS) are used to scale neural retrieval efficiently.

Flexibility: The same architecture allows **hybrid approaches** combining BM25, bi-encoders and neural cross-encoder retrievers.

Usage of Neural IR in Search Systems - Domain Adaptability

When documents and queries are represented using a neural language model (for example, BERT), the same model can be replaced with a domain-specific variant such as BioBERT or LegalBERT.

Then these specialized models are fine-tune for **topical relevance estimation** to make them "retrievers".

This enables the final retrieval system to better understand specialized terminology, abbreviations, and context-specific meanings in that domain.

For example, to design a system for medical applications, we can use a model trained on biomedical literature to generate more accurate document and query representations and fine-tune it for retrieval to do similarity estimation.

Expected benefits include:

Improved semantic matching between queries and documents.
Better handling of domain-specific synonyms (for example, "myocardial infarction" ≈ "heart attack").
Enhanced retrieval effectiveness even with limited in-domain labeled data, thanks to transfer learning.

Open Challenges and Further Readings

Neural IR – Open Challenges

Efficiency and Scalability

Neural models require large computational resources and optimized indexing to operate effectively at web scale.

Training Data Requirements

High-quality relevance annotations **(used to fine-tune models for retrieval)** are expensive and scarce, especially in domain-specific or low-resource settings.

Domain Generalization and Adaptation

Models trained on **general data** often struggle to maintain performance when transferred to new or specialized domains.

Interpretability and Explainability

Neural retrieval estimations are difficult to **interpret**, limiting trust and usability in sensitive or professional contexts.

Neural IR – Resources

An Introduction to Neural Information Retrieval: https://www.microsoft.com/en-us/research/wp-content/uploads/2017/06/fntir2018-neuralir-mitra.pdf

A Deep Look into Neural Ranking Models for Information Retrieval:
https://www.sciencedirect.com/science/article/abs/pii/S0306457319302390
https://arxiv.org/abs/1903.06902

You can also refer to the papers associated with cross and bi- encoders, or ask me for additional material.


Questions?