
A comparative study of semantic search model performance on a corpus of Legal Documents

MASTER THESIS

by
ADRIEN O'HANA

Supervisors:

Dr. Zachary SCHILLACI
Melvin KIANMANESH RAD

EPFL Supervisor:

Dr. Jean-Cédric CHAPPELIER

2022-2023

Contents

| | |
|---|-----------|
| Contents | 1 |
| 1 Introduction | 5 |
| 1.1 Motivations | 5 |
| 1.2 Objectives | 5 |
| 1.3 Outline | 6 |
| 1.4 Term clarification | 6 |
| 2 Background | 7 |
| 2.1 Information Retrieval | 7 |
| 2.2 Semantic Search | 7 |
| 2.3 Transformers | 7 |
| 2.3.1 Attention Mechanism | 8 |
| 2.3.2 Self-Attention | 9 |
| 2.3.3 Attention weights | 10 |
| 2.3.4 Pre-Training and Fine-Tuning | 11 |
| 2.3.5 Model Variants | 12 |
| 2.4 Transformers for Semantic Search | 13 |
| 2.4.1 Early Approaches | 13 |
| 2.4.2 Bi-Encoders | 14 |
| 2.4.3 Loss Functions | 15 |
| 2.4.4 Cross-Encoder Re-ranking | 17 |
| 2.4.5 Domain Adaptation | 17 |
| 2.5 Lexical Search | 19 |
| 2.5.1 Early Approaches | 19 |
| 2.5.2 BM25 Okapi | 20 |
| 2.6 Other Approaches | 21 |
| 3 Methodology | 22 |
| 3.1 Objectives and Tasks | 22 |
| 3.2 Dataset Creation | 23 |
| 3.2.1 Legal Corpus | 23 |
| 3.2.2 Query Selection | 24 |
| 3.2.3 Fine-Tuning & Evaluation Requirements | 25 |
| 3.2.4 Retrieval Span and Context | 26 |
| 3.3 IR Evaluation Metrics | 26 |
| 3.3.1 nDCG@K | 27 |
| 3.3.2 mAP@k | 27 |
| 3.3.3 MRR | 28 |
| 3.3.4 mAR@k | 28 |
| 3.4 Rater Agreement Metrics | 29 |
| 3.4.1 Spearman's rho | 29 |
| 3.4.2 Kendall's tau | 29 |
| 3.5 Annotation Campaign | 29 |
| 3.5.1 Guidelines | 30 |
| 3.5.2 Annotation Files | 30 |

| | | |
|----------|-------------------------------------|-----------|
| 3.5.3 | Three Tier Approach | 31 |
| 3.5.4 | Resulting Data | 33 |
| 4 | Results | 34 |
| 4.1 | Inter-Annotator Agreement | 34 |
| 4.2 | Domain Adaptation | 35 |
| 4.2.1 | GPL | 35 |
| 4.2.2 | AugSBERT | 36 |
| 4.3 | Comparative Analysis | 37 |
| 4.3.1 | Cross-Encoder Re-Ranking | 37 |
| 4.3.2 | Sensitivity to k | 38 |
| 4.3.3 | Binary Relevancy Results | 39 |
| 4.3.4 | Discussion | 41 |
| 5 | Conclusion and Perspectives | 43 |
| 5.1 | Key takeaways | 43 |
| 5.2 | Drawbacks and Limitations | 43 |
| 5.3 | Future work | 44 |
| | References | 45 |
| A | Queries | 50 |
| B | BERTopic | 52 |
| C | Inter-Annotator Agreement | 53 |

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Zachary, for his guidance and support throughout this research project. Zachary's knowledge in the field were critical in defining the direction of the research and in providing expert advice on technical issues.

I would also like to thank my teacher, Jean-Cédric, for his continuous support and guidance throughout the thesis. Jean-Cédric's input was invaluable in refining the direction of the research, and his regular feedback ensured that the project stayed on track.

I wish to acknowledge Melvin for his oversight and guidance especially in the early stages of the project. His motivation and encouragement throughout the research were highly valued.

Finally, I would like to thank Mauro and Prisca for funding this project. Their regular contributions insured the relevancy of the project and were critical in the successful completion of this thesis.

I would also like to extend my thanks to my colleagues, friends, and family for their support throughout the project.

Abstract

The proliferation of digital information has transformed the way we access and collect data. While the internet provides a wealth of information on every topic, retrieving relevant information from a large corpus of documents remains a challenging task, especially in specialized domains like law. The complexity and technical nature of legal documents, along with the sheer volume of information, make it difficult to search and locate relevant information using traditional keyword-matching methods.

To address this challenge, semantic search models offer a promising solution. These models go beyond simple keyword matching by considering the overall meaning of a search query and the contextual meaning of words in a searchable data space. They help to bridge the lexical gap between search queries and relevant results, providing more accurate and comprehensive results. Nonetheless, semantic search in specialized domains such as law remains challenging as the semantics used in such domains are rather specific and not the usual more general semantics.

This study aims to evaluate the effectiveness of semantic search models on a corpus of domain-specific documents and determine the best approach for retrieving relevant information on a corpus of legal documents. We explain how to construct an appropriate dataset to fit our domain-specific needs and use it to compare multiple approaches of semantic search by interpreting the appropriate information retrieval evaluation measures.

1 Introduction

1.1 Motivations

Semantic search is a way of performing information retrieval that can respond accurately to the true meaning of a user’s search query by learning from the relationship between the input words and recognising a user’s intent in order to find the appropriate search answer. As a result of the emergence of large-language models (LLMs) derived from the transformer architecture [1], there is a novel approach to semantic search: neural search.

While neural search architectures leveraging these LLMs have already been proposed, such as the sentence-transformer models [2], fine-tuning these on a specific task and domain, such as retrieving relevant passages from a corpus of legal documents, requires costly labeled data that is seldom available. A wide range of pre-trained models are accessible and can be useful in many cases; however, these models have been reported to often perform poorly in zero-shot settings [3], even worse than lexical search in some cases [4]. To the best of our knowledge, their capabilities on a homogeneous real-world use case remain undocumented.

1.2 Objectives

In this study, we are interested in discovering which techniques should be used when searching in legal documents. The results of this research will provide valuable insights into the application of semantic search models on official documents from the legal field and will also contribute to a deeper understanding of the potential and limitations of such models in other specialized domains, helping to pave the way for more effective information access and retrieval in specialized fields.

We will address the following questions:

- Which techniques can be used to encode text into a semantically meaningful data space for searching?
- How should we select a set of queries that is relevant to legal experts?
- How should the relevancy of a query-passage pair be labeled?
- Which metrics can we use to evaluate the performance of semantic search models?
- Do semantic search models that have been fine-tuned on other domains generalize well to our own, or is their performance poorer than a lexical approach?
- What are the existing techniques to perform domain adaptation of neural search models? What are their associated requirements and are they worth implementing on our domain?

1.3 Outline

In Chapter 2, we delve into the theoretical foundations of semantic search and sentence-transformers, and introduce a lexical baseline to compare semantic search models to. That chapter provides a comprehensive analysis of the relevant literature on sentence embedding techniques.

Moving on to Chapter 3, we explain our approach to constructing datasets that meet the diverse requirements for fine-tuning and evaluation. We motivate our choice of evaluation metrics and model selection and detail the methodology used for our annotation campaign. This chapter sheds light on the different challenges we encountered and the strategies we employed to overcome them.

Chapter 4 presents the results of our comparative study, highlighting the strengths and limitations of the different models we tested against our baseline. We provide a detailed analysis of the results, discussing the factors that contributed to the success of the top-performing models.

Finally, in Chapter 5, we reflect on the implications of our research and discuss the potential limitations of our study. We provide recommendations for future research, highlighting the areas where further investigation is required. By the end of this study, readers should have a clearer understanding on how to construct a dataset for semantic search evaluation and how to choose a suitable sentence-transformer model for their semantic search task on a specialized domain.

1.4 Term clarification

In this section, we clarify some important terms that will be used throughout this thesis.

- Unless specified otherwise, when we use the term **query** we are referring to a natural language search query, which is a request for information expressed in everyday language without the use of a structured language or syntax.
- In the context of this thesis, the term **document** is used as a cover term for a text of any length that is part of a given collection.
- A **sentence** is a grammatical unit of one or more words.
- A **passage** is a section or excerpt of text of considerable length, that is constructed with one or multiple sentences. It can be taken from a larger work or a collection of works, and may focus on a specific topic or idea.
- A **sentence embedding** is a vector representation of a sentence.

We often use the terms 'document', 'sentence', and 'passage' interchangeably, since sentences and passages are both documents.

2 Background

2.1 Information Retrieval

Information Retrieval (IR) entails finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections. Retrieving pertinent information from a large pool of information system resources may either rely on full-text or content-based indexing. This can involve seeking information within a document, looking for the document itself, interpreting database content or identifying metadata that contains relevant information.

Evaluation in information retrieval is heavily reliant on the nature of the problem at hand. Some evaluation metrics focus on the ordering of the first k proposed answers, which should be from most to least relevant. Other common IR evaluation metrics are the usual ones for classification: Precision, Recall and F1, which measure the quality of the top k answers disregarding their order. Evaluation is discussed in detail in section 3.3.

In this study, we focus on the retrieval of textual data. We use the term *document* as a cover term for text of any length in the given collection, and the *query* for the user input can be of any length as well. Retrieval can be done with approaches including lexical [5], sparse [6] [7], late-interaction [8] and dense [9] [10] [11], the latter being the most novel due to recent breakthroughs in deep learning applied to natural language processing (NLP).

2.2 Semantic Search

Sparse, late-interaction and dense retrieval methods are usually associated to semantic search. These methods focus on generating responses to a search query by understanding the searcher’s intent, the context of the query and the relationship between words. Such an approach can alleviate the ambiguous nature of search queries and can bridge the lexical gap that keyword based methods are often unable to overcome without heavy text preprocessing. Furthermore, identical search queries could be phrased in different ways. Semantic search also considers the relationship between words, which can be crucial to find relevant information.

Applications of semantic search include passage retrieval, duplicate question detection, fact checking, entity retrieval and question answering (QA), which can also be performed in multilingual settings. Today’s state-of the art solutions heavily rely on transformer architectures which are detailed in section 2.3.

2.3 Transformers

Recent developments, especially Transformers [1], have provided solutions better than ever before to most problems in the field of Natural Language Processing. This rather novel architecture has given rise to multiple derived models with the ability to encode semantic meaning, the most notable being BERT (Bidirectional Encoder Representations from Transformers) [12] and GPT (Generative Pre-trained Transformer) [13]. In the following paragraphs we explain the core concepts that were introduced by Devlin et al. [1]: positional encoding, self-attention and multi-head attention.

2.3.1 Attention Mechanism

”Attention” was originally introduced to improve Neural Machine Translation (NMT) [14]. In an encoder-decoder setting [1], attention is a mechanism that empowers the decoder to decide which parts of the source sentence to pay attention to. In an encoder or decoder only setting, the attention mechanism can also learn to extract meaningful representations of text for varying downstream tasks such as classification, named entity recognition, document similarity or semantic search.

The Transformer architecture was published at a time when LSTMs [15] and GRUs [16] were the state-of-the art approaches to language modeling. By dispensing with the aforementioned recurrence networks’ inherent sequential nature, transformers provided a more efficient and parallelizable architecture which relies solely on attention mechanisms.

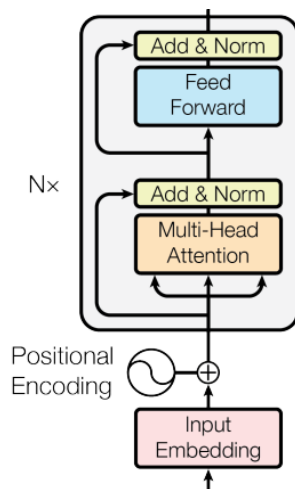


Figure 1: (picture from [1]) Encoder part of the Transformer Model architecture. $N \times$ encoder units are stacked one after the other. The top output can be trained to learn different objectives such as text classification, named entity recognition or sentence embeddings (see section 2.3.4).

BERT uses the encoder depicted in Figure 1. A text input sequence is converted to concatenated word-piece vectors¹ and attention modifies each of these initial vectors by multiplying them with context-dependent weights. This leads to a better representation of each word in the input sequence by taking into account all of the surrounding words for each word.

In order to help the encoder distinguish the position of word vectors in the input sequence, positional encodings are first added to the input embeddings. These encodings represent the order in a sequence. Just like the encoder in the original transformer paper [1], we note that by design BERT’s input is limited to 512 word-piece tokens, which is an important limitation of this model (we can not encode texts that are too long, or have to find a workaround - see section 3.2.4).

¹Most transformers like BERT represent spans of text as tensors of word-piece vectors which also gives the model the ability to represent words that are not part of its vocabulary by combining multiple word-piece vectors. Word-piece vectors will be called word vectors from now on.

2.3.2 Self-Attention

In section 2.3.1 we introduced the concept of attention and briefly described it in the context of a transformer encoder such as BERT [12]. Figure 2 depicts this process of improving word vector representations in a sequence and introduces the concept of Queries (Q), Keys (K) and Values (V). The initial word vectors are multiplied by three different learned weights to form Q, K, and V, which are the coefficients of three different linear layers, and which result in three new word vectors for each input word. Q, K and V are then combined together with a "Scaled Dot-Product Attention" to get new context-aware word vectors.

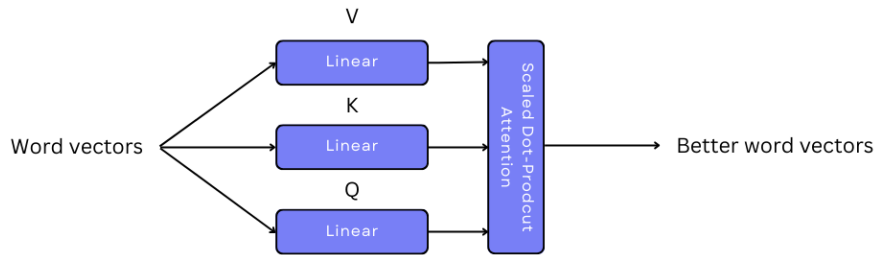


Figure 2: Single Attention head. Queries (Q), Keys (K) and Values (V) are the input word vectors combined together with a learned linear transformation to get better word representations that depend on the rest of the sequence.

As the names suggest, we can make an analogy between the attention mechanism and a retrieval system. We map a query Q with a set of key-value pairs K, V . Actually, the result of the attention function is obtained by multiplying a set of queries Q , which are representations of one of the input sequence words, with K , a different set of representations of each of the input sequence words, from which we select the (soft) maximum (Q, K) pair. This maximum pair points to the value V , which is yet another representation of each input word where the mechanism indicates to pay attention. If we add a scaling operation to the argument of the *softmax* we are using the scaled dot-product attention depicted in Figure 3.

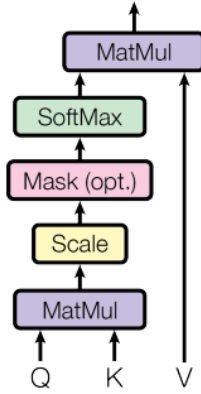


Figure 3: (picture from [1]) Scaled Dot-Product Attention. Q , K and V are the input sequence word vectors multiplied by three different weight matrices. Masking is optional².

"The input consists of queries and keys of dimension d_k , and values of dimension d_v . We compute the dot products of the query with all keys, divide each by $\sqrt{d_k}$, and apply a softmax function to obtain the weights on the values. In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix Q . The keys and values are also packed together into matrices K and V " [1]

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

We talk about "self-attention" because we are relating different positions of the input sequence in order to compute a representation of that same sequence. d_k is the size of the word piece vectors, and in the context of self-attention it is the same as d_v , since all the keys, values and queries come from the same input sequence. The output of the attention function is a tensor of context-aware vectors with the same shape as the original input.

2.3.3 Attention weights

So far we've detailed how the attention mechanism can compute better representations of a sequence, but we haven't addressed how we learn useful transformations for our input words. In practice we would like each word to attend to not only one other word, but multiple words in parallel. We use multiple attention heads (using Scaled Dot-Product Attention) which are computed side-by-side over the input vectors, concatenated together and combined with learned linear projections. This is called Multi-Head Attention.

²Masking is an optional mechanism used in various pre-training processes such as Masked Language Modeling [12] which consists in hiding different parts of an input sequence and learning to predict the missing words.

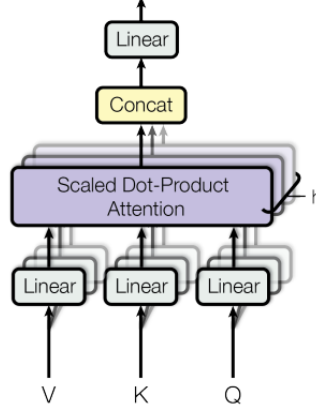


Figure 4: (picture from [1]) Multi-Head Attention.

”Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions.”

In the case of self-attention, Q , K and V are tensors of size (N, d_{model}) , N being the number of inputs and d_{model} being the length of their associated token vectors.

Multi-Head Attention:

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O$$

with

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

where $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ are the learned weights of the queries’, keys’ and values’ linear projections and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ the learned weights that combine the multiple attention heads to get the final representation. d_k and d_v are the output dimensions of the linear projections and are both equal. h is the number of attention heads.

The output of the multi-head self-attention goes into a feed-forward network (see Figure 1). Depending on the training setup, the weights of this neural network can learn to output multiple objectives. This layer introduces more non-linearity (using ReLU activation) and contain most of the learned parameters, which are trained in two stages: pre-training and fine-tuning (see section 2.3.4).

Combined with the self-attention and the feed-forward network are residual connections and layer normalization that typically prevent vanishing gradients and improve training. Multiple encoder blocks like this are stacked one after the other and used to learn sequence representations in the form of tensors of contextual dependant word vectors. The output of the encoder is a tensor of contextualized word vectors.

2.3.4 Pre-Training and Fine-Tuning

Transformers can be seen as general purpose learning units that can solve many tasks. When used in language, these models are usually pre-trained on self-supervised tasks such as Next Sentence Prediction and Masked Language Modelling (MLM). It allows the

model to learn linguistic aspects relevant to multiple target tasks by recognizing patterns in large amounts of unstructured textual data. This unsupervised pre-training provides a good task-agnostic initialization point from which to learn a more specific objective.

Using input transformations and little fine-tuning data, pre-trained large languages models can adapt to different target tasks and domains. Figure 5 shows different input transformations to use with a GPT model which allow to learn different tasks like Classification, Entailment, Similarity or Multiple Choice³.

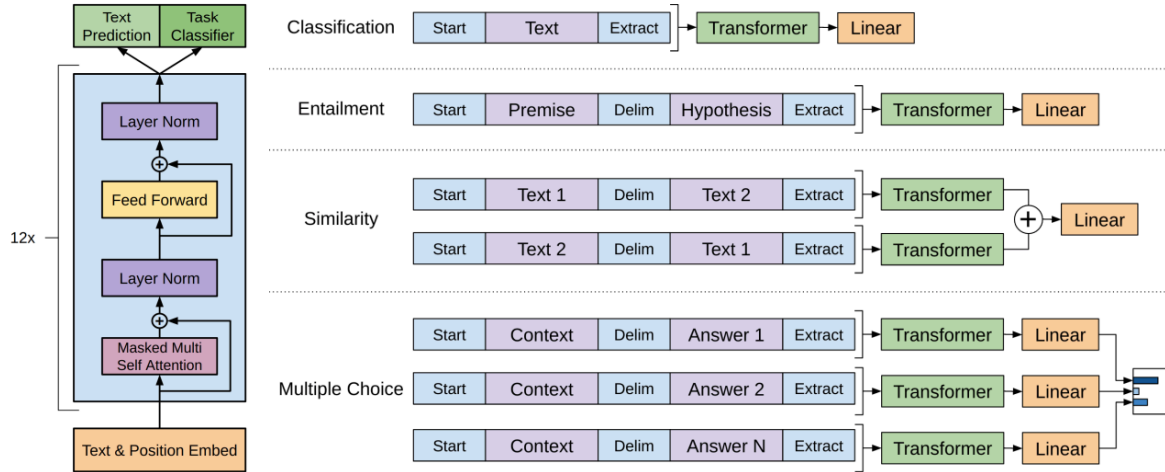


Figure 5: (picture from [13]) GPT Input transformations for fine-tuning on target tasks.

In this project we mainly focus on the encoder of the transformer architecture. The reason we don't detail the decoder is that almost all of the models studied do not use the decoder part of the architecture, which purpose is to generate text rather than encode it. In our case we are more interested in using numerical representations of text that would allow us to search efficiently through large collections. As we will see in section 2.4, there is a dedicated category of transformer encoders for semantic search.

2.3.5 Model Variants

Since the introduction of LLMs, countless alternative architectures have been developed, each building on the foundation laid by their predecessors. Some of the most notable advances have been made in the pre-training stage, with impressive enhancements introduced with models like RoBERTa [17], Electra [18], MPNet [19] and the latest iterations of GPT [20]. There are also exciting alternatives that focus on knowledge distillation, resulting in streamlined and agile networks, such as DistilBERT [21] and MiniLM [22]. With a plethora of diverse models to choose from, selecting the ideal tool for a specific task has become even more conceivable.

³Contrary to BERT that uses only the encoder part of the Transformer Architecture, GPT uses only the decoder which we haven't described. For a detailed explanation, please refer to [1]

2.4 Transformers for Semantic Search

Semantic Search is usually performed by measuring the similarity of a given query to all available textual documents, and selecting the highest scoring pairs, hoping to obtain semantically pertinent piece of texts as a result. Sentence-transformers are fine-tuned encoders that provide state-of-the-art performance for tasks ranging from question answering to clustering [23] [24]. For semantic search, their performance can be very close to the other methods on the BeIR benchmark [4].

SBERT⁴ is the name of the framework giving access to pre-trained sentence embedding models derived from a range of transformer architectures. By combining them with tools from HuggingFace⁵ and Pytorch⁶, sentence embedding models can also be created from scratch or fine-tuned with supervised and unsupervised approaches. We discuss these approaches and their use-cases in the present section.

2.4.1 Early Approaches

Initially developed to solve sentence-pair regression tasks, sentence-transformers provide two types of architectures: Cross-Encoders and Bi-Encoders.

2.4.1.1 Cross-Encoders

Cross-Encoder [25] is the name given to transformer models that have been fine-tuned to predict the degree of relevancy between two inputs, such as the similarity of two sentences, or the similarity of a query to a sentence that might answer it. These models can compute a similarity score by performing self-attention over two input sentences that have been concatenated and separated by a special token. This requires that both sentences are fed into the network, which causes a massive computational overhead and renders them inapplicable to problems such as clustering, semantic similarity, and semantic search on large corpora. As we will see in section 2.4.4, this approach remains relevant when applied to carefully selected subsets of data, because while not scalable on their own, cross-encoders provide the best accuracy [2].

2.4.1.2 Mean-pooling and the [CLS] token

Until 2019 the most common solution to solve the scalability issue while leveraging the power of large language models was to use these models' pre-trained attention layers to map spans of text to a vector space where semantically similar sentences are close to each other⁷. This would be done by one of the following schemes:

- Averaging all contextual token vectors in the the final attention layer.
- Taking the maximum vector in the the final attention layer.
- Using only BERT's "[CLS]" token to represent the sequence, which is a dedicated token inserted at the beginning of a sentence and contains sentence classification information.

⁴<https://www.sbert.net/>

⁵<https://huggingface.co/>

⁶<https://pytorch.org/>

⁷Spans of text, passages and sentences can be used interchangeably for our use case.

All three approaches result in a fixed size vector that represents the input text’s meaning.

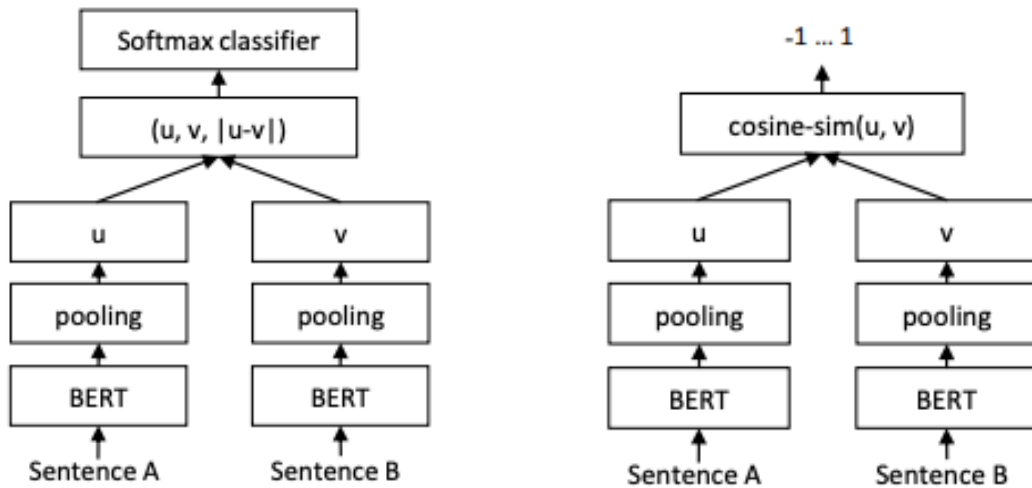
As demonstrated for Sentence-BERT [2], simply deriving a fixed representation for sentences from a pre-trained BERT model doesn’t provide good sentence embeddings, as it often achieves worst results than much simpler models like GloVe [26]⁸. To address this limitation, Reimers et al. [2] introduced bi-encoders (also called sentence-transformers). They provide semantically meaningful vector representations for spans of text derived from a range of transformer architectures.

2.4.2 Bi-Encoders

There are several sentence-transformer architectures that use different objectives and training data. In the first publication [2], the best performing models are fine-tuned in two steps starting from pre-trained BERT/RoBERTa checkpoints. Those two steps are depicted in Figure 6.

They first use a classification objective (Cross Entropy Loss) on the Natural Language Inference (NLI) data (including the Stanford NLI dataset (SNLI)[27]) which is of the form (sentence1, sentence2, class) and the classes are entailment, contradiction or neutral. The best performing sub-method found is to perform mean pooling of the contextualized token embeddings for each sentence and concatenate them in addition to the difference of the means of the two sentences.

The second uses a regression objective (mean squared error) on the Semantic Textual Similarity (STS) dataset [28] which is of the form (sentence1, sentence2, similarity score). This time a regression is performed between the cosine similarity of two sentences and their similarity score.



(a) SBERT architecture with classification objective function, e.g., for fine-tuning on SNLI dataset. The two BERT networks have tied weights (siamese network structure).

(b) SBERT architecture at inference, for example, to compute similarity scores. This architecture is also used with the regression objective function.

Figure 6: (pictures from [2]) SBERT Fine-Tuning and inference architectures.

⁸Word embedding models like GloVe, unlike transformer models, have no awareness of word ordering and do not address polysemy issues.

"We always use cosine-similarity to compare the similarity between two sentence embeddings" - or between a query and a passage in the case of semantic search. "We ran our experiments also with negative Manhattan and negative Euclidean distances as similarity measures, but the results for all approaches remained roughly the same." [2]

2.4.3 Loss Functions

After the first publication [2], several better performing architectures were proposed on their website⁹. Modifications to the original architecture include the introduction of new loss functions, which can determine how well the embedding model will perform on a specific downstream task. The biggest factor in determining the type of loss to use is the shape of the training data.

The original losses introduced were the Softmax Loss (classification objective) and the Cosine Similarity Loss (regression objective) which have already been presented in figure 6. Newer loss functions used in this study include the Multiple Negative Rankings Loss [29] and the Margin MSE Loss [30].

Some vectors resulting from these methods are normalized, while others are not. It depends if the similarity measure in the loss function is a cosine similarity or a dot-product (dot-product models have a tendency to exhibit superior performance, albeit at the expense of lower computational speed).

2.4.3.1 MSE and Cross-entropy

Let's first define the commonly used Mean Squared Error (MSE) Loss and Cross-Entropy Loss functions.

The MSE loss is commonly used as the objective function in linear regression. It is defined as:

$$L_{MSE} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (1)$$

where y_i is the actual value of the target variable, \hat{y}_i is the predicted value, and N is the number of data points in the dataset.

Cross-entropy loss is a type of loss function used in machine learning and deep learning to measure the difference between predicted probability distribution and the true probability distribution of a classification problem. It is commonly used in multi-class classification problems and is defined as follows:

$$L_{CE} = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^C y_{ij} \log(\hat{y}_{ij}) + (1 - y_{ij}) \log(1 - \hat{y}_{ij}) \quad (2)$$

where N is the number of samples in the dataset, C is the number of classes, y_{ij} is a binary indicator (0 or 1) for whether sample i belongs to class j and \hat{y}_{ij} is the predicted probability that sample i belongs to class j .

In the context of sentence-transformers (see section 2.4.2), these two common losses can be used with different inputs and targets that are usually constructed using one or

⁹<https://www.sbert.net/>

multiple pairs of sentence embeddings and their associated target similarity scores or class labels.

2.4.3.2 Softmax Loss

We implement the Softmax loss by concatenating the sentence embeddings u_i and v_i with the element-wise difference $|u_i - v_i|$, multiplying it with the trainable weight $W_t \in \mathbb{R}^{3n \times k}$ and using the Softmax function:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^C e^{z_k}} \quad (3)$$

The loss to minimize is then the Cross-entropy loss L_{CE} (2) with the following inputs:

$$\hat{y}_{ij} = \sigma(W_t(u_i, v_i, |u_i - v_i|))_j \quad (4)$$

$$y_{ij} = \begin{cases} 1 & \text{if pair } (u_i, v_i) \text{ belongs to class } j \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where n is the dimension of the sentence embeddings, k the number of labels, and $j \in C$ corresponds to one of C classes.

2.4.3.3 Cosine Similarity Loss

The Cosine Similarity Loss is none other than the MSE Loss L_{MSE} (1) using the cosine of the input pairs' embeddings to predict its similarity score prediction \hat{y}_i . The target y_i is the "true" similarity label of the input pairs, between 0 and 1.

$$\hat{y}_i = \cos(u_i, v_i) \quad (6)$$

2.4.3.4 Margin MSE Loss

This loss was introduced for knowledge distillation and has been used to fine-tune the current top performing SBERT models for semantic search available on their website. For a given query Q , we have a positive and a negative answer, respectively P^+ and P^- , and train the model to learn from the difference between the positive and negative similarity scores with the query.

We compute the MSE loss L_{MSE} (1) with the following target and prediction:

$$\hat{y}_i = |sim(Q, P^+) - sim(Q, P^-)| \quad (7)$$

$$y_i = |gold_sim(Q, P^+) - gold_sim(Q, P^-)| \quad (8)$$

"We train ranking models on batches containing triples of queries Q , relevant passages P^+ , and non-relevant passages P^- . We utilize the output margin of the teacher model labels as label to optimize the weights of the student model." [30]

By default for sentence-transformers, the similarity function $sim()$ is the dot-product between a query embedding and a passage embedding and $gold_sim()$ is the "true" similarity which could be measured by a cross-encoder.

2.4.3.5 Multiple Negatives Ranking Loss

According to the SBERT website, the Multiple Negatives Ranking Loss is a great loss function if you only have positive pairs, for example, only pairs of similar texts like pairs of paraphrases, pairs of duplicate questions, pairs of (query, response), or pairs of (source_language, target_language). The input is a list of sentence pairs (a_i, p_j) with only one positive example (p_i) and $n - 1$ negative examples $(p_j, j \neq i)$. It then uses these labels with the Softmax Loss, minimizing the the Cross-entropy loss L_{CE} (2) for softmax normalized scores [29].

2.4.4 Cross-Encoder Re-ranking

As mentioned in section 2.4.1.1, cross-encoders can not be used to perform semantic search because the amount of computation needed to find the highest similarity scores between a query and all available documents is too long. We can still make use of cross-encoders in a scalable manner if we first use a much faster approach to retrieve the top K results - such as the dense and lexical approaches introduced in sections 2.4.2 and 2.5.2 - and use a cross-encoder to re-rank these results.

This usually improves the quality of the very first few results. For example, if we want to find the top 5 results to a query in a collection of 1'000'000 documents, we can first retrieve the 100 highest scoring documents with a bi-encoder and then use a cross-encoder to find the top 5 results out of these 100, which means there are only 100 heavy computations instead of 1'000'000.

2.4.5 Domain Adaptation

In this section we briefly introduce the three state-of-the-art domain adaptation techniques for pre-training and fine-tuning sentence-transformers in unsupervised and supervised ways.

2.4.5.1 TSDAE

Transformer-based Sequential Denoising Auto-Encoder [31] is an unsupervised domain adaptation technique used as a pre-training step. It outperforms the previous pre-training approach introduced with BERT called Masked Language Modeling (MLM) [12]. Noise is added to the input sequences by deleting tokens, encoding them into sentence embeddings and teaching a decoder to reconstruct the original input from the damaged embedding. Instead of mean pooling to get the resulting sentence embedding, we can use the [CLS] token in this case. This approach is similar to MLM [12] but the reconstruction must be done from a single sentence vector as opposed to the full attention tensor of the sentence, which forces better sentence representations in practice.

2.4.5.2 GPL

Generative Pseudo Labeling [3] is an unsupervised domain adaptation technique for fine-tuning sentence-transformers. At the time of publication (April 2022) this approach outperformed previous state-of-the-art dense retrieval methods on the Information Retrieval Benchmark (BeIR) [4]. GPL consists of the three following steps (see Figure 7):

1. Query Generation: Generate synthetic queries on passages from the target corpus using a pre-trained query generation model (T5 [32]);

2. Negative Mining: Find similar passages to each query generated in step 1 using pre-trained dense retrievers;
3. Pseudo Labeling: Assign similarity scores to all the generated pairs using a pre-trained cross-encoder, and use these pseudo-labels to fine-tune a sentence-transformer.

As we’ve seen in section 2.4.1.1, cross-encoders provide much higher accuracy for similarity scores than bi-encoders. These accurate labels can be used to train a bi-encoder. At the end of these three steps, we’ve created a new pseudo-labeled dataset of query answer pairs, usually with multiple queries for the same passage and because of negative mining also multiple passages per query. We use the similarity scores from the cross-encoder in order to fine-tune a new sentence-transformer on our generated dataset using the Margin MSE Loss (see section 2.4.3.4).

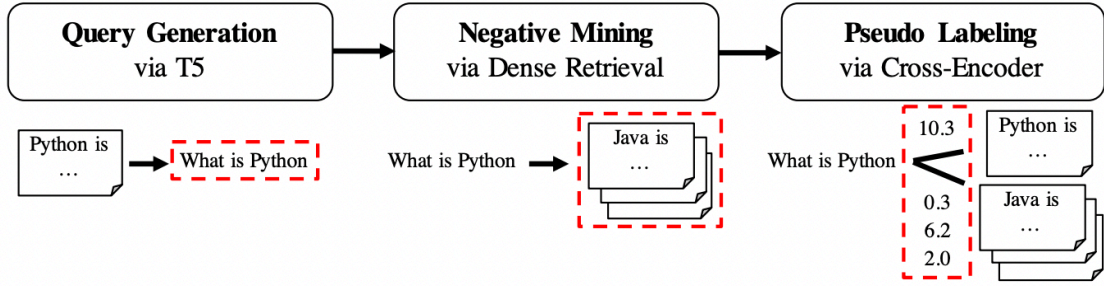


Figure 7: (picture from [3]) Generative Pseudo Labeling (GPL) for training domain-adapted dense retriever.

2.4.5.3 Augmented SBERT

Augmented SBERT [33] is a supervised domain adaptation technique that allows to fine-tune sentence-transformers with small amounts of labeled data. Similarly to GPL, a cross-encoder is used to annotate an augmented dataset which in turn is used to fine-tune a bi-encoder (see section 2.4.2). The training process starts with a *gold dataset* of labeled examples. This gold dataset is small and it is not enough to fine-tune a bi-encoder, so we need to augment it.

The approach consists of the following steps (see Figure 8):

1. Fine-tune a cross-encoder using the gold dataset (cross-encoder need less fine-tuning data than bi-encoders);
2. Augment the data to a *silver dataset* by adding new unlabeled pairs; Thakur et al. [33] propose multiple approaches for augmentation, the simplest being to sample random pairs from the gold data, creating new sentence pairs;
3. Label the silver data using the fine-tuned cross-encoder in the first step, and fine-tune a sentence-transformer using the pseudo labeled silver data combined with the original gold data.

Similarly to the pseudo-labeling step in GPL, we are using the similarity scores from a cross-encoder in order to fine-tune a new sentence-transformer on the silver dataset using the Cosine Similarity Loss (see section 2.4.3.3).

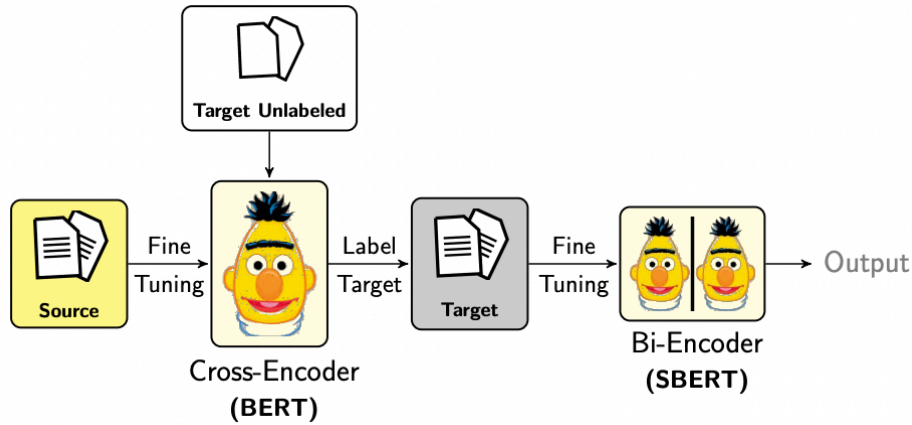


Figure 8: (picture from [33]) Domain adaptation with AugSBERT.

2.5 Lexical Search

As we’ve introduced in section 2.1, there are multiple approaches to performing IR with text data. In section 2.4 we’ve seen that bi-encoders can provide dense embedding representations that leverage attention to encode semantic meaning, and we can use them to perform semantic search. In order to measure the performance of such dense vector search models we typically use a lexical model as a baseline for comparison.

2.5.1 Early Approaches

2.5.1.1 Bag of Words

Before word vectors were introduced, a common approach to encode a document was to use a Bag of Words representation [34]. To represent a document, we use a vector of the length of the indexing vocabulary in our corpus (one dimension per indexing word). Such a vector has in each dimension the count of appearances in the document of its associated vocabulary word. This is the same as applying one-hot encoding to each indexing word and summing their one-hot-encoding representations to represent a document. From this representation we can build classifiers or implement a simple lexical search.

2.5.1.2 TF-IDF

A better way to represent documents is to modify the word counts by taking into account their frequency in the entire corpus. If a word exists in all documents, then it is less likely that it will help in differentiating documents. Similarly, if a word exists only in a given document it should be a determining factor in the representation of that document.

TF-IDF stands for term frequency- inverse document frequency and is intended to reflect how important a word is to a document in a collection of documents.

The term frequency of term t in a document d is:

$$tf(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \quad (9)$$

where $f_{t,d}$ is the raw count of term t in document d , i.e., the number of times that term t occurs in document d .

The inverse document frequency is:

$$idf(t) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \quad (10)$$

where $|D|$ is the total number of documents in the corpus D , and $df(t) = |\{d \in D : t \in d\}|$ is the number of documents where the term t appears.

Finally the tf-idf value for each term t in the document d is:

$$tfidf(t, d) = tf(t, d) \cdot idf(t) \quad (11)$$

If we want to use this to score the relevancy of document d for query q we need to sum the tf-idf scores of each of the N terms q_i in that query.

$$tfidf(q, d) = \sum_{i=1}^N tf(q_i, d) \cdot idf(q_i) \quad (12)$$

2.5.2 BM25 Okapi

We introduce BM25 Okapi [5], which is a modification of the TF-IDF representation specifically intended for information retrieval. With TF-IDF, if a document has a relevant word appearing twice as much as another, it is considered twice as relevant, which is usually undesirable in an IR setting. For example if we are looking for the word "data", a document including the word 100 times could be as relevant as a document including that word 200 times. Instead of TF increasing linearly with the number of occurrences, we would rather like TF to increase quickly with the first few word occurrences and to almost stop increasing once the count of occurrences gets saturated. Another factor to take into account is the length of the document, if a document has very few words and contains one relevant word, it should be as relevant as a document containing many words and many times that relevant word.

With iterations over the years, many variants taking into account document length and term frequency saturation have emerged, one of the most famous being BM25. We choose the values of $k = 1.5$ and $b = 0.75$ considered "*reasonably good in many circumstances*" [5].

$$BM25(d, q) = \sum_{i=1}^N idf_{BM25}(q_i) \cdot \frac{tf(q_i, d) \cdot (k + 1)}{tf(q_i, d) + k \cdot (1 - b + b \cdot \frac{|d|}{L})} \quad (13)$$

where

- q_i : i -th term of the query q
- N : the number of terms in the query q
- $|d|$: number of words in document d

- $tf(q_i, d)$: the frequency of term q_i in document d
- L : the average length of documents in the collection
- k : term frequency parameter
- b : length normalization parameter

and

$$idf_{BM25}(q_i) = \log \frac{N - df(q_i) + 0,5}{df(q_i) + 0,5} \quad (14)$$

where

- $df(q_i)$: total number of documents that contain the term q_i
- N_{docs} : total number of documents

Many other modifications can improve the performance of lexical approaches [35] which we will not detail nor implement in this study. These include stopwords removal, stemming and lemmatization and n-gram tokenization. There are also some later improvements to BM25 that render it more suitable to some settings [36], such as using the title of documents in order to select a first relevant subset on which to perform BM25.

2.6 Other Approaches

We’ve introduced sentence embeddings with SBERT and lexical approaches such as BM25. There are other state-of-the-art approaches none of which achieve the best performance across most datasets [4]. ColBERT [8] for example is a late-interaction model that recently beat the best sentence-transformer (GPL) on the IR Benchmark (BeIR) with their second version [37]. Because its implementation at the time of writing is less straightforward and because it is a more memory intensive method, we decided not to study this model. Another approach worth mentioning is to use OpenAI¹⁰ which as of recently reported on their website an even better average nDCG@10 than ColBERTv2 or GPL on the BeIR dataset with their *text-embedding-ada-002* model. Because they are not open source, and not free we also decided not to include these embeddings in our study. In addition, they have been reported to perform worse than sentence-transformers in the past [38]. Since sentence-transformer models are open source, achieve very good performances on various tasks backed by a series of publications and are accompanied by a framework allowing domain adaptation and giving access to powerful pre-trained models, all semantic search models evaluated in this study are sentence-transformers.

¹⁰<https://platform.openai.com/docs/guides/embeddings/embedding-models>

3 Methodology

3.1 Objectives and Tasks

Considering the objectives initially established in section 1.2 and the background on information retrieval and semantic search provided in section 2, we outline the scientific methodology we will follow to reach these objectives.

We want to compare the performance of a range of information retrieval approaches in the legal domain. Typically IR is evaluated with a set of queries and a set of documents, we therefore choose to create a dataset of legal documents that will allow us to measure and compare the performance of these models using standard IR evaluation metrics. For a full discussion on evaluation metrics, see section 3.3.

The selection of approaches to compare is a collection of sentence-transformers presented in 2.4 as well as the lexical baseline introduced in 2.5.2. Part of the dataset will also be used for supervised domain adaptation, which should be evaluated on a different set of query-passage pairs.

The objectives of this study can be re-formulated with the following tasks:

1. Select a collection of documents containing textual information of interest to legal experts.
2. Devise a selection of queries of interest to legal experts and with relevant information contained in the data.
3. Construct a dataset using a range of top performing pre-trained sentence-transformers for semantic search and a lexical baseline, and conduct an annotation campaign. These annotations should be used to evaluate the performance of the models used to construct the dataset, and to fine-tune a new model using a supervised domain adaptation approach.
4. Apply the three domain adaptation approaches presented in section 2.4.5 using our collection of documents and queries and gather more annotations and queries from the new models. These newer annotated query-passage pairs should be used to evaluate the newly introduced domain adapted models, and to evaluate some of the initial models on a larger query set.
5. Compare the performance of all selected information retrieval approaches employing standard IR evaluation metrics for the purpose of identifying which approaches are worth pursuing in a specialized domain like law.

An example of a chosen legal act can be seen in Figure 9, and a query-passage pair in Figure 10.

3.2 Dataset Creation

We present here the procedure to create a dataset representative of our legal experts' interests, starting from a publicly available collection of legal texts. This dataset should allow us to evaluate and compare multiple semantic search models, and draw insightful conclusions as to which approaches are most relevant to the legal domain and in a real-world use case.

We use expert judgement to select multiple search queries and to annotate the degree of relevancy of numerous model answers to these queries. Part of the annotated data should also be used in an active learning step to evaluate the supervised fine-tuning approach presented in section 2.4.5.3.

3.2.1 Legal Corpus

The data we will be using is a large corpus of legal texts from the European Union (EUR-Lex¹¹, see Figure 9). It contains more than 10'000 Legal Texts (including Jurisprudence (court decisions) and Legislation (laws)) translated in multiple languages including English and can be easily extracted from the web. The texts contain arguments on topics that are varied ranging from science and agriculture to customs and finance with answers to many queries. Sometimes these can be tremendously specific, for example about a protein contained in a chemical process, or they can have a much broader sense, such as the definition of "property".

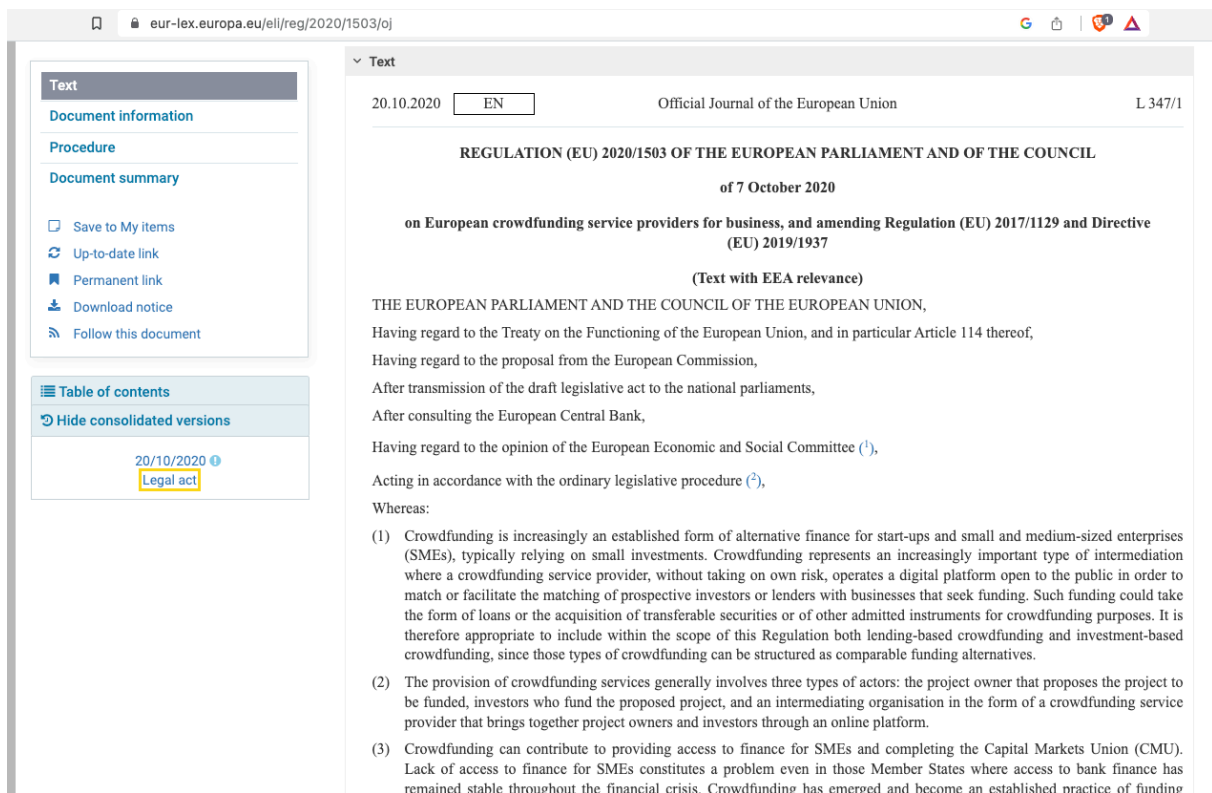
The image is a screenshot of the EUR-Lex website. The browser's address bar shows the URL 'eur-lex.europa.eu/eli/reg/2020/1503/oj'. On the left side, there is a sidebar with a 'Text' tab selected. Below it, there are links for 'Document information', 'Procedure', and 'Document summary'. Further down, there are icons for 'Save to My items', 'Up-to-date link', 'Permanent link', 'Download notice', and 'Follow this document'. At the bottom of the sidebar, there is a 'Table of contents' section with a link to 'Hide consolidated versions'. The main content area shows the details of a legal act. At the top, it says '20.10.2020' and 'EN' (English). The title is 'REGULATION (EU) 2020/1503 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 7 October 2020 on European crowdfunding service providers for business, and amending Regulation (EU) 2017/1129 and Directive (EU) 2019/1937'. Below the title, it says '(Text with EEA relevance)'. The text of the regulation begins with 'THE EUROPEAN PARLIAMENT AND THE COUNCIL OF THE EUROPEAN UNION, Having regard to the Treaty on the Functioning of the European Union, and in particular Article 114 thereof, Having regard to the proposal from the European Commission, After transmission of the draft legislative act to the national parliaments, After consulting the European Central Bank, Having regard to the opinion of the European Economic and Social Committee (1), Acting in accordance with the ordinary legislative procedure (2), Whereas:'. There are three numbered paragraphs: (1) Crowdfunding is increasingly an established form of alternative finance for start-ups and small and medium-sized enterprises (SMEs), typically relying on small investments. Crowdfunding represents an increasingly important type of intermediation where a crowdfunding service provider, without taking on own risk, operates a digital platform open to the public in order to match or facilitate the matching of prospective investors or lenders with businesses that seek funding. Such funding could take the form of loans or the acquisition of transferable securities or of other admitted instruments for crowdfunding purposes. It is therefore appropriate to include within the scope of this Regulation both lending-based crowdfunding and investment-based crowdfunding, since those types of crowdfunding can be structured as comparable funding alternatives. (2) The provision of crowdfunding services generally involves three types of actors: the project owner that proposes the project to be funded, investors who fund the proposed project, and an intermediating organisation in the form of a crowdfunding service provider that brings together project owners and investors through an online platform. (3) Crowdfunding can contribute to providing access to finance for SMEs and completing the Capital Markets Union (CMU). Lack of access to finance for SMEs constitutes a problem even in those Member States where access to bank finance has remained stable throughout the financial crisis. Crowdfunding has emerged and become an established practice of funding.

Figure 9: A legal act part of the text collection from EUR-Lex.

¹¹<https://eur-lex.europa.eu/homepage.html>

3.2.2 Query Selection

We’ve established in section 3.1 that we wanted to compare the performance of different IR techniques on a collection of documents. Accordingly we need a set of queries large enough to accurately compare their performances. With the help of the two legal experts that have continuously collaborated with us on this project, we select queries that we consider interesting from a legal standpoint.

3.2.2.1 Requirements

There are two major requirements that guided the process of selecting a set of queries to evaluate our models on. First of all, the selected queries must be *relevant* to our corpus of documents and ideally discuss topics that are addressed multiple times. Without this requirement we might end up with a labeled dataset of mostly irrelevant pairs, which would not only prevent us from fine-tuning a model but also restrain our ability to evaluate and compare the other models. A second requirement is that we need a large enough set of queries, and as suggested by Christopher Manning in "Introduction to Information Retrieval" [39], *"As a rule of thumb, 50 information needs has usually been found to be a sufficient minimum"*.

3.2.2.2 Query Selection using Topic Modeling

In pursuance of creating queries with a guarantee that the topic they address is discussed, we first implemented Topic Modeling using the default BERTopic approach [24]. We embed all sentences of our corpus with a pre-trained sentence-transformer model trained for semantic similarity. Then we perform dimension reduction using UMAP [40] so we can apply a clustering algorithm to the sentence embeddings¹². Clustering is done with HDBSCAN [41] and we then represent each cluster of sentences with the top tf-idf scoring words across all cluster sentences. The tf-idf scores are class based, meaning the entire cluster of sentences is considered as the document. In this case, we make the hypothesis that clusters can be considered as topics. The first few topics with the most sentences represented by their top c-tf-idf scoring words can be found in Appendix B.

Topic Modeling allowed us to explore the dataset and its contents and initiated the discussion on how to select a set of interesting queries for our study. We can already see a varied range of topics mentioned in the documents, such as electricity, insecticides, alcohol, anti-dumping... And we can easily imagine to write some search queries associated to the found topics, such as a definition query.

With such an approach there are two major caveats to consider. First, creating queries only relevant to the largest clusters would not be representative of useful legal information needs, we would therefore have to select topics at random on which we would create associated queries. Also, only using the top tf-idf words in our queries not only biases the lexical gap between query and documents but it also leads to queries that are too simple and of little interest to end users searching for information across the corpus. We concluded by establishing that detailed search queries would allow to more easily evaluate the degree of relevancy of a relevant document as opposed to a simple keyword search where relevancy would be binary.

¹²The dimension reduction being memory intensive we’ve had to reduce the data by keeping only texts since 2018 which is little over 300k sentences.

We can imagine looking at more than one representative word for each cluster, for example in topic 28 of Appendix B, the words "imposing", "definitive", "anti", "dumping", "china" are the most frequent in that cluster. Combined together we could form a more elaborate query, such as "Which imports originating from china have had anti-dumping duties imposed?". The issue with creating more complex queries from looking at topic modeling results is that we can not know if the answer exists in the corpus, which once again might have led us to annotate mostly irrelevant documents. For all these reasons we decided to use a query generation model instead.

3.2.2.3 Generative Model

The next step we considered to create a set of queries to evaluate is using the T5[32] query generation model proposed in the GPL approach [3]. The model has been trained on the MSMARCO [42] dataset, a large IR dataset containing anonymized queries asked through the Bing search engine, and human generated answers for each of these. With the help of legal experts we select a subset of these randomly sampled queries that covers an interesting range of information needs containing legal acronyms, definition, and explanation queries. With this approach, each query has been generated on an existing passage of the data which partially ensures there is at least one good answer that the search models can find. Browsing and selecting queries generated by the T5 model also gives valuable insight into the GPL method and its learning ability on our corpus. Furthermore, we need many queries that cover a wide range and this is the most efficient method to find them. The selected queries can be found in Appendix A.

Here's an example of a query and part of the passage it was generated from:

What does a reference laboratory do?

Verification of performance and compliance with common specifications or with other solutions chosen by the manufacturer 1. For the purposes of the task referred to in Article 100(2), point (a), of Regulation (EU) 2017/746, [...] 1. EU reference laboratories shall verify the performance of the device and its compliance with common specifications or with other solutions chosen by the manufacturer as set out in paragraph 1 based on the results of the laboratory tests referred to in paragraph 2. 4. EU reference laboratories shall provide their opinion within 60 days after the latest of the following dates: (a) the date of signature of the contract referred to in Article 10(1), point (a), by all contracting parties; (b) the date of receipt of all the necessary documentation and information from the notified body as referred to in Article 11, paragraph 2, and clarifications referred to in Article 11, paragraph 3; (c) the date of receipt of equipment from and completion of any training by the manufacturer as referred to in Article 11, paragraph 4; (d) the date of receipt of the samples of the device to be tested. The opinion of the EU reference laboratories shall be detailed and shall provide reasons for the conclusions and recommendations made.

3.2.3 Fine-Tuning & Evaluation Requirements

We would like to create a dataset that will be used for fine-tuning with Augmented SBERT [33], which should therefore be at least in the range of 2000-60000 labeled example pairs for the gold dataset. This in combination with the 50 queries rule of thumb means we need at least 40 annotated answers per query.

We also need to annotate query/answer pairs with a label indicating the degree of relevancy of the answer to the associated query. One way of implementing this is to use labels between 0 and 5 like in the STS datasets [28]. As suggested in [33], we can convert the labels to a continuous similarity score between 0 and 1 by dividing the labels by 5, which can then be interpreted as the similarity between the two associated embeddings.

Finally, most of the previously mentioned IR related publications use a specific IR evaluation measure, the nDCG@10. This measure is described in section 3.3 but already mentioned here as it is an evaluation on the top 10 results for a given query. This implies we need at least 10 results per query and per model if we also want to report the nDCG@10.

3.2.4 Retrieval Span and Context

As we’ve seen in section 2.3, the input of the original transformer was limited to 512 tokens, and this has stayed the case with many encoder based transformer architectures such as BERT, leading to an important limitation of dense vector search over other approaches: we can only encode texts that have less word-piece tokens than 512. The size of the passages from our legal documents we are searching through should therefore be limited to 400 words which roughly corresponds to the input token limit of the English transformer models used, 512 tokens.

One has to decide how to separate these passages. This could be to separate them in paragraphs if they are not too long on average, unfortunately the data is not structured enough for robust paragraph extraction and there were other priorities. Another option would be simply to split the legal texts into groups of sentences containing less than 400 words, but there is a risk that an answer will be separated in two different groups.

We experimented with including passages of varying lengths starting from every sentence but the semantic search models would mostly output overlapping passages in this situation instead of selecting different answers. Finally we came to the conclusion that splitting the texts in sentences was a good approach, as long as the assessors have access to the surrounding context of that sentence. Sentence segmentation is done using the dependency parsing implementation [43] [44] from spaCy [45].

3.3 IR Evaluation Metrics

We would like to evaluate the ranking of a set of retrieved passages given a query and aggregate this evaluation between multiple queries. For this we usually need the ideal ranking of a retrieval system’s answers. An issue that arises is that while an IR system might be excellent at ranking its first K proposed results, it can fail to provide the best possible answer from the document collection (which is much larger than K). In some TREC¹³ datasets for example (Text REtrieval Conference), only the most 100 relevant answers to each query are annotated, and this can have harmful effects on evaluation [46].

When evaluating the first K answers of such a system in a realistically large document set, it is impossible to collect exhaustive relevance judgements, so we might not know what is the real best possible answer. Also called the ”IR recall problem” [47], it complicates proper comparison of IR metrics applied on diverse datasets and even renders some metrics generally unused, such as Recall. People usually select Precision@10 or nDCG@10.

¹³<https://trec.nist.gov/>

Since we are mostly interested in finding relevant results, we also need to compare some measure of the quality of the very few first answers, regardless of their order. A measure that indicates if we have been able to find what we were looking for in those first documents or not. For example if one model always outputs K top scoring answers when another mostly outputs mediocre K answers, it is easy to choose.

Annotations that indicate the degree of relevancy on a scale from 0 to 5 can be used to calculate all the following evaluation metrics. Those will be used to report, interpret and compare the performance of our models in section 4. Binary judgements have more traditionally been used. Conversion from $[0, 5]$ to $\{0, 1\}$ can be easily done with a threshold at 3: all relevancy judgements above 3 included are converted to 1 and all below are converted to 0. This loosely aligns with our annotation guidelines (see section 3.5.1) considering that relevancy is a subjective matter and that a score of 3 is assigned when the topic of the query is discussed.

3.3.1 nDCG@K

Today, the most widely used metric to evaluate textual IR is the normalised discounted cumulative gain ($nDCG$) on the K top results retrieved. As opposed to other order-aware metrics such as $mAP@K$ or MRR that use binary relevancy annotations, $nDCG@K$ can make use of ordinal relevancy labels rel_k just like the 0 to 5 range we've decided to use.

The cumulative gain $CG@K$ is the sum of the relevancy scores rel_k from the first to the K -th result. The discounted cumulative $DCG@K$ gain uses the cumulative gain $CG@K$ and a logarithmic discount $\log(1 + k)$ that makes it order-aware.

$$CG@K = \sum_{k=1}^K rel_k \quad (15)$$

$$DCG@K = \sum_{k=1}^K \frac{rel_k}{\log(1 + k)} \quad (16)$$

$$nDCG@K = \frac{DCG@K}{iDCG@K} \quad (17)$$

Finally, the ideal Discounted Cumulative Gain $iDCG@K$ is the same discounted sum using the ideal relevancy scores of that sequence if we picked the best possible ordering of answers existing in our corpus. We divide the DCG by the ideal DCG for normalization purposes. We note the calculation of the $iDCG@K$ is highly dependent on the distribution of the available relevance judgements which are often partial. We generally consider all unavailable relevance judgements as irrelevant which still allows us to use the $nDCG@K$ for comparison purposes.

3.3.2 mAP@k

Another widely used metric is $P@K$, the Precision on the first K results with binary relevancy.

$$brel_k = \begin{cases} 1 & \text{if item at the } k_{th} \text{ rank is relevant} \\ 0 & \text{otherwise} \end{cases} \quad (18)$$

P@K can be defined as the fraction of relevant items in the top K recommended items, or the proportion of relevant items amongst retrieved items.

$$P@K = \frac{N_K}{K} \quad (19)$$

with $N_K = \sum_{k=1}^K brel_k$ being the number of relevant documents for a given query in the first K results.

Some queries might only have 5 relevant results while others have more than 20. To compensate for this variability across queries we use the mean average precision score. The $mAP@K$ is a mean of the average precision values $AP@K$ for each query with the $AP@K$ itself being the average between $P@1$, $P@2$, ..., $P@K$. While this measure is less intuitive it has been shown to be more stable and informative than simply averaging P@K [48].

$$AP@K = \frac{1}{N_K} \sum_{k=1}^K P@k \cdot brel_k \quad (20)$$

To obtain the $mAP@K$ we only average the precision at relevant k values.

$$mAP@K = \frac{1}{Q} \sum_{q=1}^Q AP@K(q) \quad (21)$$

where Q is the number of queries and $AP@K(q)$ is the $AP@K$ of the retrieved documents for query q .

3.3.3 MRR

Another commonly used metric in Information Retrieval is the Mean Reciprocal Rank (MRR) which measures the quality of the ranking in terms of the rank of the first relevant document retrieved. MRR is particularly useful in scenarios where users are interested in finding only one relevant document. It only takes into account the rank of the first relevant document and ignores the ranks of subsequent relevant documents, which can be important in certain scenarios.

The MRR is calculated as follows:

$$MRR = \frac{1}{Q} \sum_{q=1}^Q \frac{1}{rank_q} \quad (22)$$

where Q is the total number of queries, and $rank_q$ is the rank of the first relevant document retrieved for query q . If no relevant documents are retrieved for a particular query, the corresponding $rank_q$ is set to infinity.

3.3.4 mAR@k

Perhaps the most intuitive evaluation measure utilizing our ordinal relevancy scores and with significant usefulness for comparison is the mean of the average relevancy score of the first K answers of a model, that we will call $mAR@K$. We first compute the average relevancies up to K for each query and then compute the mean of these average relevancies across all queries.

$$AR@K = \frac{1}{K} \sum_{k=1}^K rel_k \quad (23)$$

$$mAR@K = \frac{1}{Q} \sum_{q=1}^Q AR@K(q) \quad (24)$$

where again Q is the number of queries, $AR@K(q)$ is the average relevancy $AR@K$ for query q and rel_k is the discrete relevancy score of answer k between 0 and 5.

3.4 Rater Agreement Metrics

There are multiple ways to assess the agreement between two raters. For ordinal variables like our discrete values from 0 to 5 we typically use Spearman or Kendall’s rank correlation coefficients.

3.4.1 Spearman’s rho

Spearman’s rank correlation coefficient ρ_s is often used to compare rankings. It is none other than a correlation coefficient computed on the ranks of two series of observations.

$$\rho_s = 1 - \frac{6 \sum_{i=1}^n d_i^2}{n(n^2 - 1)} \quad (25)$$

where d_i is the difference between the two ranks of each observation and n is the total number of observations.

3.4.2 Kendall’s tau

Kendall’s rank correlation coefficient τ_A measures the strength of association of two rankings.

$$\tau_A = \frac{n_c - n_d}{n_0} \quad (26)$$

where $n_0 = n(n - 1)/2$ is the number of pairs of observations to compare between two rankings, n_c is the number of concordant pairs, n_d is the number of discordant pairs and n refers to the total number of observations in each ranking.

3.5 Annotation Campaign

Creating a fully annotated dataset that accurately matches the distribution of the real dataset is a challenging task that requires annotating many low-scoring answers. However, the cost of annotation can quickly become prohibitive, making it essential to balance the cost of annotation with the value of the annotated data. Additionally, we wish to evaluate the performance of a supervised fine-tuned model on new annotated data. We should avoid annotating too many negative answers for queries with a significant number of positive answers, allowing us to focus on obtaining enough fine-tuning ability.

We contract our assessors via the Upwork platform¹⁴. Two people have been carefully selected in a pool of candidates that all have a legal background, mostly paralegals with a

¹⁴<https://www.upwork.com/>

degree in law. The other criteria for selection were mostly based on cost and availability. Semantic search can be very subjective and sometimes hard to solve, especially when having to annotate on a grid between 0 and 5. To have a measure of how tricky some search results can be to judge, we used the two annotators to rate every query-passage pair.

3.5.1 Guidelines

Here are the annotation guidelines we provided to our selected annotators.

Your job is to rate answers between 0 and 5 (5 is a very good answer and 0 is completely irrelevant).

Please keep in mind that the goal is to find a relevant passage in a fixed corpus of legal documents (here all European court decisions since 2018). We want to evaluate if the tool points the user in the right direction given a search query.

Most importantly: Don't overthink, it should be your judgment. But here are a few guidelines to give an idea.

- If the answer contains a reference to an article that seems to contain the answer: 5
- If the answer is a detailed description or an overall description of the answer: 5
- If the answer is partial: 4
- If the answer is implied: 4
- If the right topic is discussed but the answer is not contained: 3
- If a close topic is discussed but the answer is not contained: 2
- If the answer is unrelated: 1
- If the answer doesn't make sense at all: 0

Then followed a series of instructions on how to use the app you can see a screenshot of in Figure 10. They can select questions one by one and annotate all their answers.

3.5.2 Annotation Files

We give a *.json* file containing the first fifteen answers from multiple models to different queries and provide a url with a custom annotation application where an assessor can rate the proposed sentences between 0 and 5 while seeing their context and title.

Different files are generated three times in the duration of the project. For every file generated, we use multiple models to retrieve the top 15 results of different models on varying queries. Generating these files for annotation efficiently requires maintaining indices to go back and forth between the text embeddings (which are computed beforehand), their corresponding passages, the document titles, the context of the passages, and the relevancy score that has been annotated. It also requires to format back and forth the annotations and manage the duplicate query-answer pairs that multiple models have provided. Details on these files are reported in section 3.5.3. The general approach to generate is depicted in Figure 11.

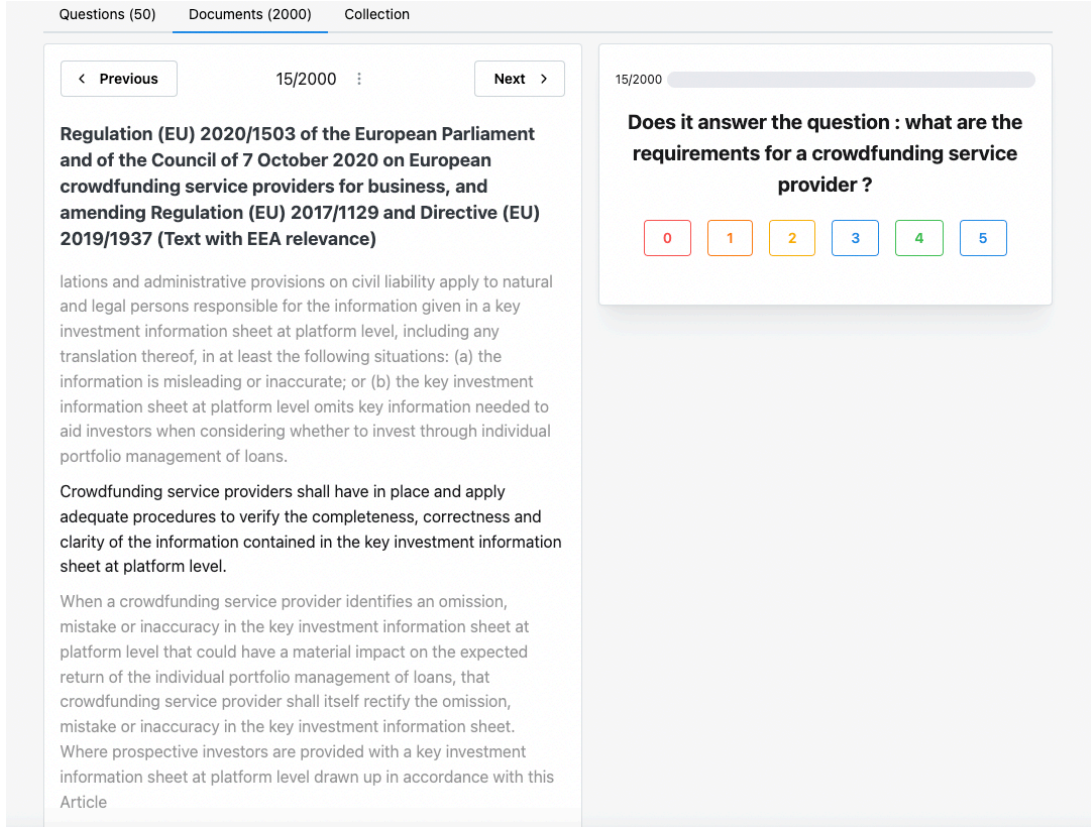


Figure 10: Example of a relevant passage from EUR-Lex shown with its context and title on the annotations application. They can rate from 0 to 5 and it automatically goes on to the next task to annotate.

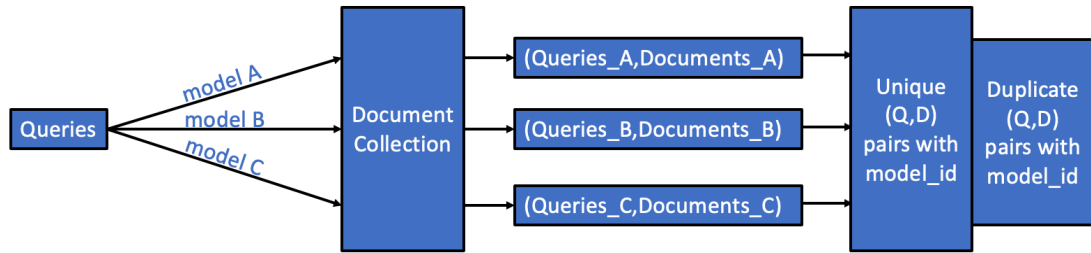


Figure 11: Flowchart: How annotation files with unique pairs are generated for a set of queries and a given document collection. After the passages for each query and each model have been retrieved, we combine the different model pairs back together and keep the duplicate pairs aside. All this is done while keeping track of which model retrieved each answer (model_id), but also additional data not depicted here, such as the document information (title, id, surrounding text) and the score and ranking attributed by each model.

3.5.3 Three Tier Approach

The approach to simultaneously construct a dataset and evaluate and fine-tune some models is loosely planned from the start and adjusted along the way. We first want to

make sure we have a look at the data and select a handful of queries, then that the annotation set up is running as expected and that the results of the very first selections are encouraging. We can then design experiments that contribute to a proper evaluation of some of the best existing solutions. We mainly have two datasets, one for fine-tuning, and one for final evaluation only. The former is used in a supervised task with Augmented SBERT from section 2.4.5.3 and an unsupervised task with GPL to evaluate the effect of the number of training steps on its performance. The latter is used to evaluate these two fine-tuning models, and is used in combination with the first dataset to evaluate the other four models we enumerate in 3.5.3.1.

3.5.3.1 Round 1: Model Selection

We select three top performing pre-trained semantic search sentence-transformers which have BERT-base, DistilBERT or MPNet architectures fine-tuned on multiple datasets. These can be found on the sentence-transformers repository in HuggingFace.

- The *msmarco-**bert**-base-dot-v5* and *msmarco-**distilbert**-dot-v5* have been trained on 500K QA pairs from the MS MARCO dataset [42](Bing search queries).
- *the multi-qa-**mpnet**-base-dot-v1* is trained on 215M QA pairs from: WikiAnswers, PAQ, Stack Exchange, MS MARCO, GOOQA, Amazon-QA, Yahoo Answers, SearchQA, ELI5, Quora Question Triplets, Natural Questions, SQuAD2.0, TriviaQA.

We also implement a *BM25* search with default parameters $b = 0.75$ and $k = 1.5$. This round is intended to make sure the annotation guidelines are correct and the application is working as expected. It is also to have a first idea of performance from different models on our own data. We annotated the first 15 answers from the 4 models above for 15 different queries. We also implement re-ranking of the top 15 answers by using a top performing cross-encoder, the *cross-encoder/ms-marco-MiniLM-L-12-v2*, also fine-tuned on the MS MARCO dataset.

3.5.3.2 Round 2: Active Learning

Once the framework to annotate and the models have been tested, we gather more information with these same models, adding some GPL model answers to the annotations and now a total of 50 queries, including the first 15. The GPL parameters and process are detailed in section 4.2.1. We additionally adjust the initial guidelines using the raters' feedbacks, ensuring the clarity of their job¹⁵ before the next round.

3.5.3.3 Round 3: Evaluation

The data from the previous round is used to select the best number of training steps for GPL. We also experiment with using longer non-overlapping passages that we construct by joining groups of sentences together, which lead to better queries overall and a higher performance. Round 2's data is also used as the gold data to fine-tune using the Augmented SBERT method (2.4.5.3).

¹⁵In the first round of annotations, by looking closely at the annotations with most disagreements between the two raters we were able to identify that one of them thought he had to rate the retrieved passages in their entirety, even if they contained a perfect answer.

3.5.4 Resulting Data

We will refer to the resulting annotated query answer pairs from round 2 as (Q_2, D_2) and from round 3 as (Q_3, D_3) . When combined using only the passages retrieved by models that we applied for the construction of both datasets, namely the three pre-trained sentence-transformers and BM25, we refer to the dataset as (Q_{23}, D_{23}) . The results in section 4 will always concern one of the three settings.

D_2 was generated on the first set of 50 queries Q_2 , using 5 different models and their top 15 answers per query, resulting in 3750 annotated query-passage pairs. Since multiple models provide the same answers we remove duplicates before annotation and propagate back the labels afterwards for evaluation. Duplicates removal is done by considering a passage as well as its surrounding context which the annotator has access to. (Q_2, D_2) actually has only 2411 unique query-passage pairs.

Similarly, D_3 has been created using 50 other queries Q_3 and the 15 top answers for each query using 6 different models. They are the same models as the ones used to create D_2 except for the GPL model, which we improved in between (discussed in section 4.2.1). The 6th model is the AugSBERT model that was trained on (Q_2, D_2) and should only be tested on (Q_3, D_2) . This results in 4500 annotated query-passage pairs out of which 3072 are unique.

Finally, (Q_{23}, D_{23}) was generated using the four starting models on the first set of 50 queries Q_2 ¹⁶ and the same four on query set Q_3 , resulting in 6000 annotated query-passage pairs for 100 queries. Examples of query-passage pairs can be found in Table 1.

| model_id | k | query | passage | round |
|----------------------------|---|---|--|-------|
| multi-qa-mpnet-base-dot-v1 | 5 | What are the macroeconomic indicators? | The macroeconomic indicators are: production, production capacity, capacity utilisation, sales volume, market share, growth, employment, productivity, magnitude of the amount of countervailable subsidies, and recovery from past dumping or subsidisation. | 2 |
| msmarco-bert-base-dot-v5 | 4 | What measures should hosting service providers take to prevent terrorism? | Hosting service providers shall set out clearly in their terms and conditions their policy for addressing the dissemination of terrorist content, including, where appropriate, a meaningful explanation of the functioning of specific measures, including, where applicable, the use of automated tools. | 3 |

Table 1: Examples of query-passage pairs generated in Rounds 2 and 3. Both pairs belong to (Q_{23}, D_{23}) , the first from Round 2 belongs to (Q_2, D_2) and the second from Round 3 belongs to (Q_3, D_3) .

¹⁶There were five models in round 2, but we’ve discarded the answers from the first GPL model. It can not be evaluated on the new set a queries because we haven’t used it in the next round, so its answers have not been annotated.

4 Results

In this chapter, we present results using our two collected datasets (see section 3.5.4) in various ways. We gain insight on the level of agreement of our two annotators for the studied task (see section 3.5), present our training results of the domain adaptation techniques for sentence-transformers, and provide two tables of binary and ordinal relevancy IR evaluation metrics.

First we explore the results from the annotation protocol and present measures of inter-annotator agreement. Second we describe the training process results for our two domain adaptation methods. Finally, we compare the performance of all presented approaches with the support of various evaluation metrics.

4.1 Inter-Annotator Agreement

In Figures 12 and 13 we show the distribution of relevancy scores given by two different annotators for the same task. These are the relevancy scores for the 15 top answers of 4 different models to 100 different queries (Q_{23}, D_{23}). The domain adapted models are not shown as they can only be assessed on the passages retrieved with the last 50 queries (Q_3, D_3).

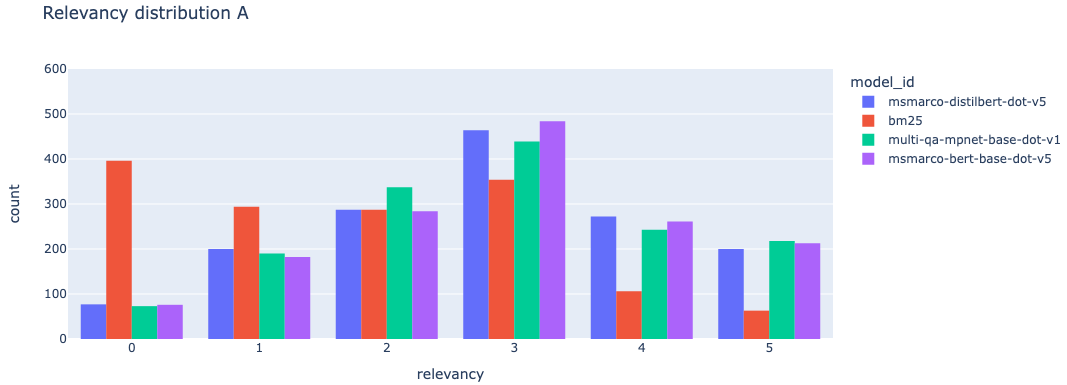


Figure 12: Judgements of rater A on four models evaluated on 100 queries (Q_{23}, D_{23})

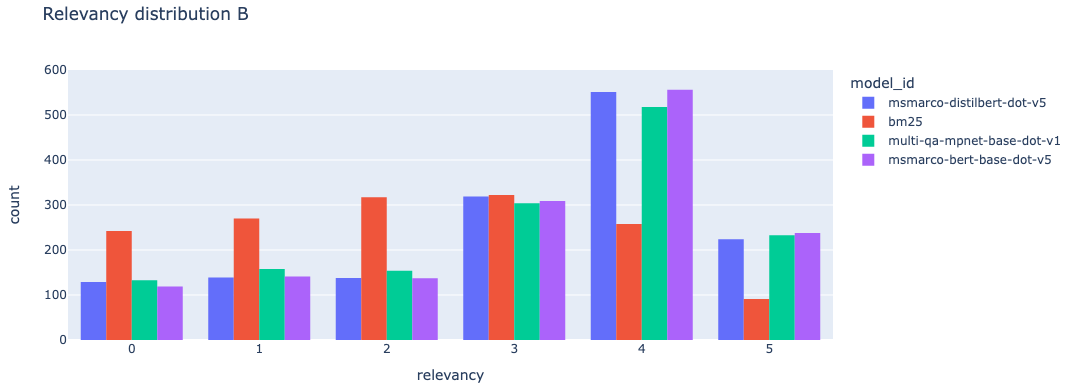


Figure 13: Judgements of rater B on four models evaluated on 100 queries (Q_{23}, D_{23})

For the data presented here, the rater agreement metrics are $\tau_A = 0.375$ and $\rho_s = 0.465$. These rank correlation measures are low but they still indicate a moderate correlation between the two raters. At first glance it seems that the major differences between both annotators are due to a shift of 1 in the relevancy assessment.

In all the following results, we obtain our gold labels both for evaluation and supervised fine-tuning by averaging the labels from our two annotators. We discard any query-passage pair where the disagreement is strictly larger than 2 which we consider as outliers. On the data presented above these strong disagreements amount to 11% of the data. For a more detailed analysis of rater disagreements see Appendix C.

4.2 Domain Adaptation

4.2.1 GPL

Following the recommendations from the GPL publication (see section 2.4.5.2), we initialize our model with a DistilBERT model that has already been trained on MSMARCO data with the Margin MSE Loss (see section 2.4.3.4). The rest of the parameters and model choices are also the same as in the publication [3]: a DocT5Query model trained for query generation on the MSMARCO dataset, using nucleus sampling with temperature 1.0, $k = 25$ and $p = 0$, two DistilBERT and MiniLM dense retrievers with cosine-similarity trained on MSMARCO from Sentence-Transformers, and for pseudo labeling, we use a cross-encoder trained on MSMARCO. Fine-tuning for 100k steps takes about 14 hours with batch size 32 on a single NVIDIA A100 GPU. Evaluation on the first annotated 50 queries (Q_2, D_2) allows to select the best performing GPL model at 140k steps of which the results on the 50 other queries (Q_3, D_3) are reported in Tables 3 and 4.

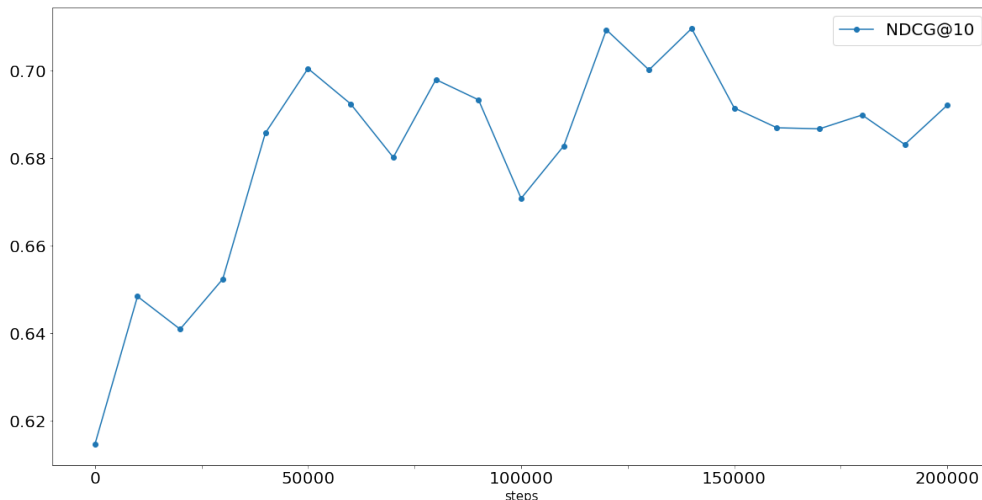


Figure 14: Average nDCG@10 on the 50 first queries (Q_2, D_2) for different training steps. The performance of GPL plateaus after 100k steps.

Since it has been reported to result in even better performance, we also experimented with TSDAE, the pre-training approach presented in section 2.4.5.1. Evaluation on our first 50 queries (Q_2, D_2) showed a decrease in the $nDCG@10$ whether we trained for 1 or

10 epochs. Figure 15 suggests TSDAE pre-training should lead to a better $nDCG@10$ from the start of GPL training, and there is little reason to believe a lower performing initial checkpoint could lead to better results. Additionally we implemented the original TSDAE approach with a BERT-base¹⁷ model instead of a DistilBERT model.

To apply TSDAE pre-training on a DistilBERT model instead we would need to implement extra steps which are not provided by the sentence-transformers library. This is due to the fact that the original DistilBERT did not provide support for being used as a decoder. For these reasons we do not include our results for TSDAE in this study.

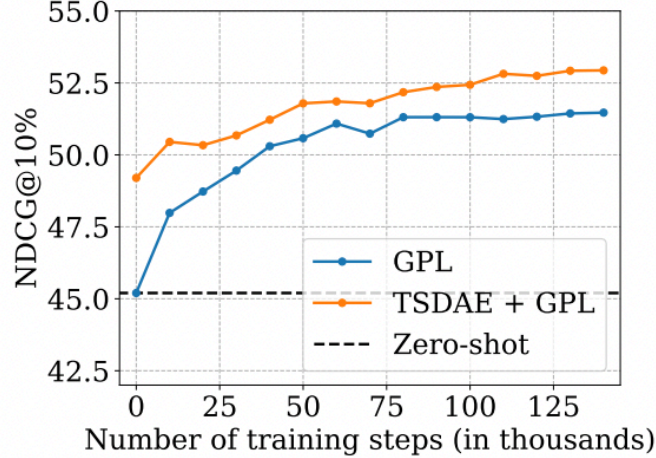


Figure 15: (picture from [3]) Influence of the number training steps on the averaged performance. The performance of GPL begins to be saturated after 100K steps. TSDAE helps improve the performance during the whole training stage.

4.2.2 AugSBERT

We fine-tune a sentence-transformer model using the Augmented SBERT approach presented in section 2.4.5.3. Our gold dataset is the set of 2411 unique labeled query-passage pairs (Q_2, D_2) , from which we’ve dropped duplicates based on the passage only this time, instead of the passage and its context (see section 3.5.4). We actually have 2075 unique pairs for training, with only 50 unique queries.

In the first step we fine-tune a cross-encoder with our gold data starting from a *bert-base-uncased* checkpoint during 1 epoch. We then create a larger silver dataset by including all possible combinations of query-passage in the gold data. Thirdly, we use our fine-tuned cross-encoder to label all the newly created pairs in our silver dataset. After duplicates removal our silver dataset has 11’681 query-passage pairs out of which 9606 have been labeled by the cross-encoder. Finally, we use the silver dataset to fine-tune a sentence-transformer, again starting from *bert-base-uncased* and training for 1 epoch.

We verify that the fine-tuning has improved the performance of our cross-encoder and sentence-transformer on the training set by calculating the Spearman rank correlations¹⁸ between the gold labels and the predictions, which is reported in Table 2. A better

¹⁷BERT exists in different sizes, a smaller size often performs worse but inference is much faster.

¹⁸We’ve introduced the Spearman rank correlation in section 3.4.1 in the context of rater agreement. It is also commonly used to compare rankings of different IR approaches, just like in the original SBERT publication [2].

performance of the fine-tuned bi-encoder over the fine-tuned cross-encoder already suggests that we might have over-fitted the set of queries.

| Model | pre-trained | fine-tuned |
|---------------|-------------|------------|
| cross-encoder | 0.66 | 0.82 |
| bi-encoder | 0.57 | 0.86 |

Table 2: Spearman rank correlation with the gold labels from (Q_2, D_2) . First comparison between our fine-tuned cross-encoder to a top performing cross-encoder trained on the MSMARCO dataset [42], the *ms-marco-MiniLM-L-12-v2*. Second comparison between our fine-tuned sentence-transformer to one of the top performing pre-trained sentence-transformers we’ve introduced in section 3.5.3.1, the *multi-qa-mpnet-base-dot-v1*.

4.3 Comparative Analysis

4.3.1 Cross-Encoder Re-Ranking

To re-rank our models, we use a top performing cross-encoder that has been trained on the MSMARCO dataset: *ms-marco-MiniLM-L-12-v2*. We calculate the cross-encoder scores for all the previously annotated query-answer pairs of each model and re-rank the passages accordingly. Using a cross-encoder in such a fashion - on only 15 results that have first been retrieved - remains a scalable approach as it takes less than one second on average using a laptop CPU. The results in Table 3 show that re-ranking almost consistently improves the $nDCG@10$ for all the models, but the $mAR@10$, showing the true quality of the top answers, does not really benefit from it. The ordering is always improved but the average relevancy suffers from this reordering for half the model answers. We also report plot the data from Table 3 in Figure 16 where we can visually interpret the standard deviations of the metrics for each model across all queries.

| Model | nDCG@10 | mAR@10 |
|--------------------------------------|------------------------|------------------------|
| multi-qa-mpnet-base-dot-v1 | 0.88 \pm 0.09 | 2.98 \pm 1.42 |
| msmarco-bert-base-dot-v5 | 0.86 \pm 0.10 | 3.01 \pm 1.38 |
| msmarco-distilbert-dot-v5 | 0.85 \pm 0.12 | 2.94 \pm 1.43 |
| gpl | 0.86 \pm 0.13 | 2.91 \pm 1.35 |
| bm25 | 0.72 \pm 0.21 | 1.85 \pm 1.41 |
| augsbert | 0.61 \pm 0.28 | 1.55 \pm 1.40 |
| multi-qa-mpnet-base-dot-v1-ce | 0.91 \pm 0.07 | 3.01 \pm 1.38 |
| msmarco-bert-base-dot-v5-ce | 0.89 \pm 0.08 | 2.97 \pm 1.40 |
| msmarco-distilbert-dot-v5-ce | 0.88 \pm 0.10 | 2.92 \pm 1.42 |
| gpl-ce | 0.89 \pm 0.10 | 2.86 \pm 1.35 |
| bm25-ce | 0.84 \pm 0.17 | 1.95 \pm 1.40 |
| augsbert-ce | 0.78 \pm 0.27 | 1.71 \pm 1.42 |

Table 3: IR evaluation metrics @10 based on the ordinal relevancy scores on the last 50 queries (Q_3, D_3) . Models ending with “-ce” have had their top 15 answers for each query re-ranked with a cross-encoder.

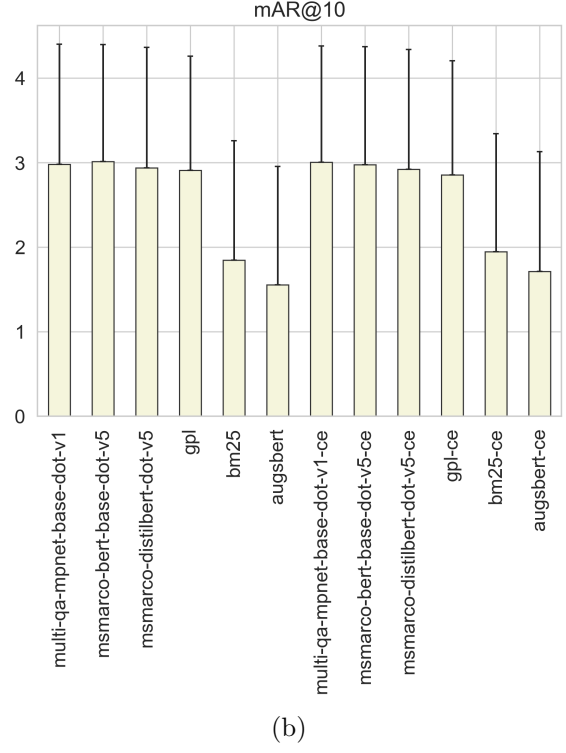
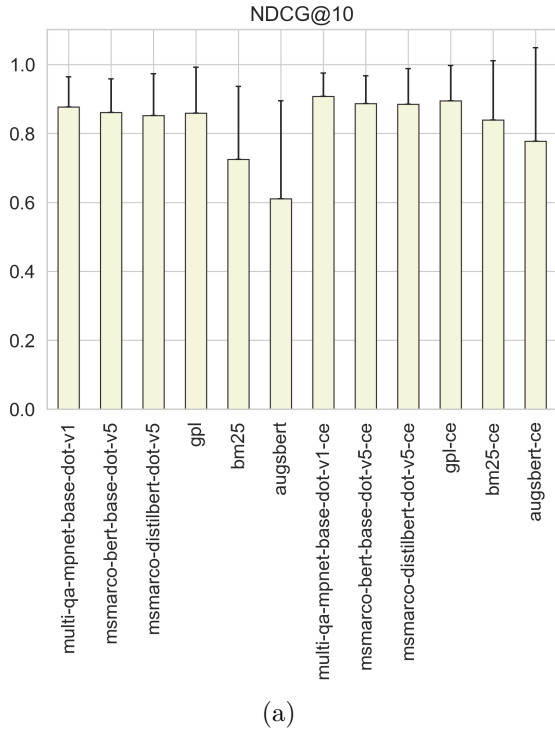


Figure 16: Bar-plot of ordinal relevancy evaluation metrics with standard deviation error-bars. Data from Table 3.

4.3.2 Sensitivity to k

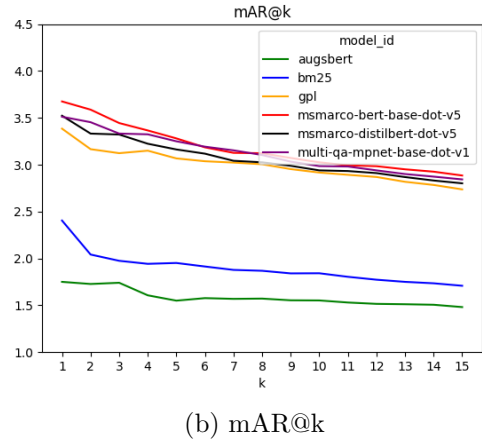
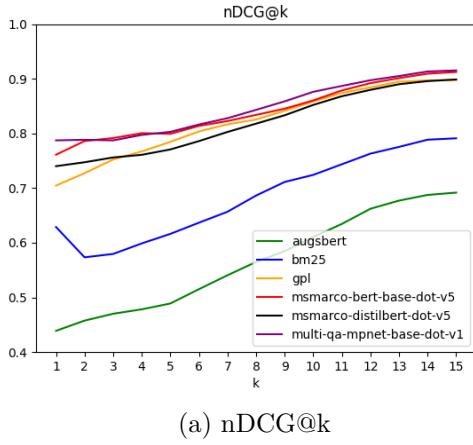
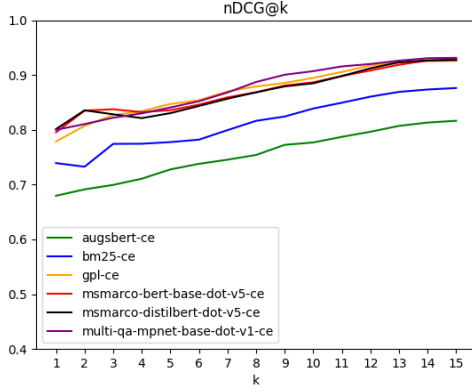
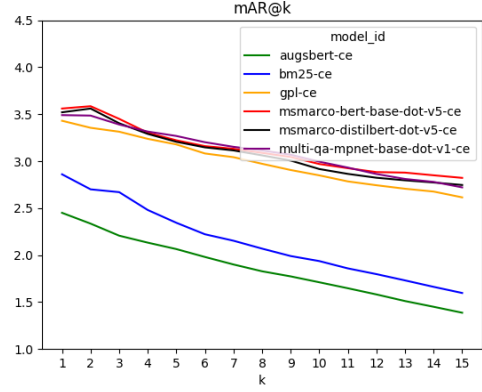


Figure 17: Evaluation results on the passages retrieved from the 50 queries (Q_3, D_3). Results averaged across queries for each model.



(a) nDCG@k



(b) mAR@k

Figure 18: Evaluation results on the passages retrieved from the 50 queries (Q_{3,D_3}). Results averaged across queries for each model. All models have had their top 15 answers per query re-ranked with a cross-encoder.

We want to make sure that reporting the results for $k = 10$ like most other IR studies remains a reasonable choice. We can see that in most situations there is only a small variation in the model performance ranking depending on k . The models are almost parallel across k which means it is fair to select 10, where there is little variance, to distinguish between them.

4.3.3 Binary Relevancy Results

Here we present more results on (Q_3, D_3), this time using the threshold of 3 to convert our labels to binary, as we’ve introduced in section 3.3. This allows to report different metrics, such as a different $nDCG@10$ than the one that used ordinal variables, $P@10$, $mAP@10$ and the MRR . Again the results in Table 4 are plotted in Figures 19 and 20.

| Model | nDCG@10 | P@10 | mAP@10 | MRR |
|-------------------------------|-------------------|-------------------|-------------------|-------------------|
| multi-qa-mpnet-base-dot-v1 | 0.74 ±0.24 | 0.56 ±0.48 | 0.64 ±0.24 | 0.85 ±0.30 |
| msmarco-bert-base-dot-v5 | 0.76 ±0.22 | 0.51 ±0.48 | 0.65 ±0.25 | 0.83 ±0.30 |
| msmarco-distilbert-dot-v5 | 0.72 ±0.25 | 0.54 ±0.49 | 0.62 ±0.27 | 0.85 ±0.27 |
| gpl | 0.69 ±0.28 | 0.36 ±0.49 | 0.59 ±0.32 | 0.81 ±0.33 |
| bm25 | 0.46 ±0.32 | 0.20 ±0.46 | 0.34 ±0.28 | 0.65 ±0.28 |
| augsbert | 0.43 ±0.38 | 0.19 ±0.42 | 0.33 ±0.32 | 0.55 ±0.35 |
| multi-qa-mpnet-base-dot-v1-ce | 0.73 ±0.27 | 0.59 ±0.36 | 0.65 ±0.30 | 0.85 ±0.28 |
| msmarco-bert-base-dot-v5-ce | 0.73 ±0.25 | 0.64 ±0.48 | 0.65 ±0.34 | 0.83 ±0.30 |
| msmarco-distilbert-dot-v5-ce | 0.74 ±0.20 | 0.52 ±0.49 | 0.63 ±0.31 | 0.85 ±0.27 |
| gpl-ce | 0.75 ±0.27 | 0.53 ±0.49 | 0.60 ±0.31 | 0.81 ±0.33 |
| bm25-ce | 0.51 ±0.33 | 0.32 ±0.47 | 0.35 ±0.28 | 0.65 ±0.42 |
| augsbert-ce | 0.43 ±0.35 | 0.24 ±0.45 | 0.30 ±0.32 | 0.55 ±0.45 |

Table 4: IR evaluation metrics @10 based on the binary relevancy scores on the last 50 queries (Q_3, D_3). Models ending with “-ce” have had their top 15 answers for each query re-ranked with a cross-encoder.

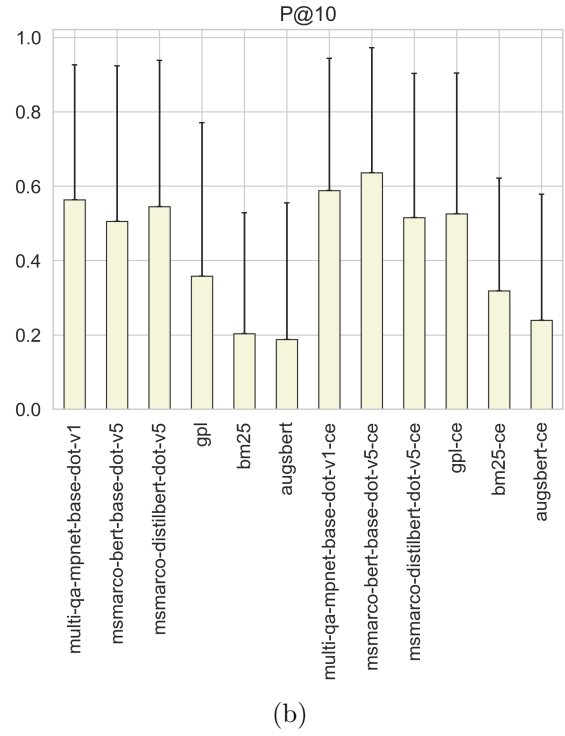
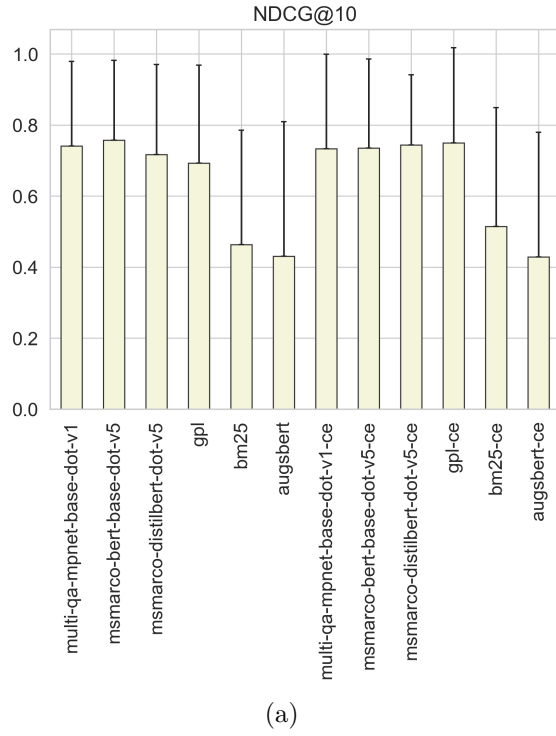


Figure 19: Bar-plot of binary relevancy evaluation metrics with standard deviation error-bars. Data from Table 4.

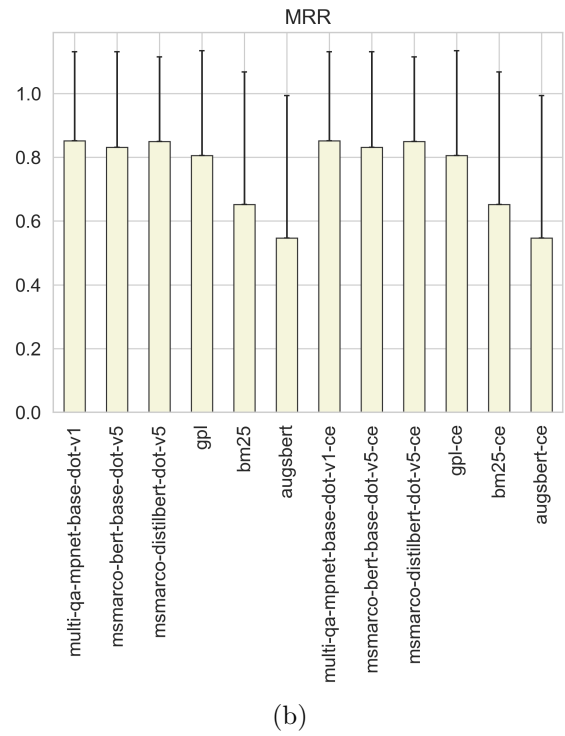
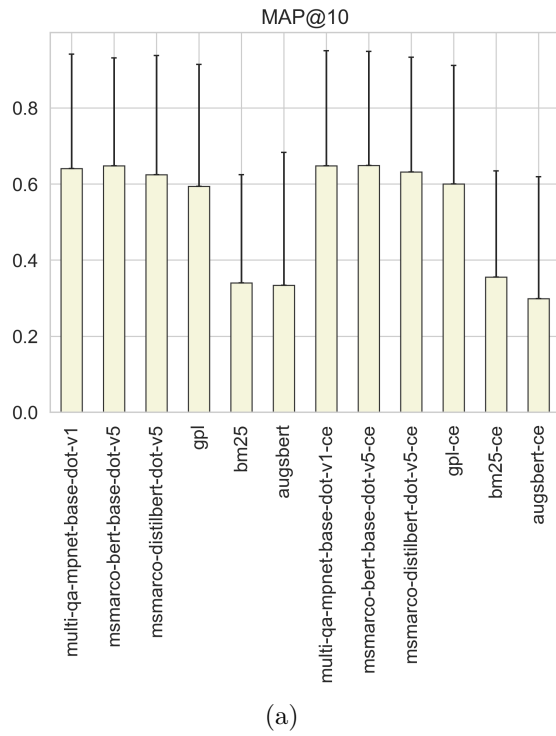


Figure 20: Bar-plot of binary relevancy evaluation metrics with standard deviation error-bars. Data from Table 4.

4.3.4 Discussion

Considering the $nDCG@10$ results reported in Table 3 and Figure 16, we can confidently say that cross-encoder re-ranking consistently improves the order of the top 10 results. Regarding the quality of the top 10 answers, the $mAR@10$ does not seem to be consistently improved (nor worsened). If we want to verify this observation using the binary relevancy results in Table 4 and Figures 19 and 20, the $nDCG@10$ does not indicate much and the MRR remains the same, but the $mAP@10$ is almost consistently improved. With this information we can reasonably assume that cross-encoder re-ranking always improves the ranking of all the selected IR techniques on legal documents. The slight discrepancies between the two tables suggest the information provided by ordinal labels can be a determining factor in model comparisons.

Again looking at Tables 3 and 4, we can compare the different model performances. First of all, we see that the top performing models are the three pre-trained MPNet, BERT and DistilBERT sentence-transformers, with BERT being almost consistently better than the DistilBERT model. Furthermore, the distinction between the performance of the MPNet model (trained on multiple datasets including MSMARCO) and the BERT model (trained only on MSMARCO) is subtle. Since binary results lead to slightly different deductions, we consider that the three models have similar performances.

The supervised domain adaptation using Augmented SBERT leads to the worst performing model in all scenarios. There are many reasons that could explain why fine-tuning on (Q_2, D_2) did not lead to a good performing model on a new set of queries. It is likely that we need a larger set of queries to start with. Originally [33], the gold data contains unique pairs of similar sentences, but we’ve used the same queries with different passages multiple times. Additionally, we augment the dataset with this same set of queries and the model probably over-fits these 50 queries. This could explain the significant better performance obtained on the training dataset after fine-tuning (reported in Table 2) compared to the poor performance reported on the evaluation dataset (Q_3, D_3) . An example of the improved performance on one query from the training data is shown in Table 5.

The other domain adaptation technique we’ve implemented, GPL, has an overall good performance, but does not beat the ordering or the quality of the passages retrieved by any of the pre-trained sentence-transformers. This indicates the technique might be worthwhile in out-domain settings where pre-trained models do not perform so well, if we consider it’s been trained in a completely unsupervised way and starting from a BERT-base checkpoint.

Finally, we expected the combination of BM25 and cross-encoder re-ranking to outperform all other approaches, but BM25 has the second to last performance. This is most probably due to the fact that re-ranking is only done in a very scalable way on only the first 15 results, as opposed to re-ranking a larger set of documents. We also used a very simple implementation of BM25 which can be improved with preprocessing steps and parameter tuning.

| k | pre-trained | fine-tuned |
|-----|---|--|
| 1 | In addition, according to the Court (17), the purpose of investigations conducted in accordance with Article 23 of the basic Regulation is to ensure the effectiveness of anti-dumping duties and to prevent their circumvention. | The Commission recalled that the purpose of imposing anti-dumping measures is to restore a level playing field so that Union producers and third country producers compete on a level playing field. |
| 2 | In addition, according to the Court (18), the purpose of investigations conducted in accordance with Article 13 of the basic Regulation is to ensure the effectiveness of anti-dumping duties and to prevent their circumvention. | The Commission recalled that the objective of anti-dumping measures is to eliminate unfairly priced imports from the Union market in order to re-establish a level playing field for all market participants. |
| 3 | Thus, having evidence tending to show continuation of dumping is sufficient to trigger an investigation on whether there is continuation or recurrence of dumping. | It should be recalled that the purpose of the anti-dumping measures is not to prevent imports, but to restore fair trade and ensure that imports are not made at dumped and injurious prices. |
| 4 | Rather, the objective of an investigation pursuant to Article 11(3) of the basic Regulation is to determine whether there is a lasting change of circumstances that warrants the re-calculation of the anti-dumping duty for the applicant. | The Commission noted that the aim of the anti-dumping measure is to re-establish fair competition in the Union market. |
| 5 | The investigation therefore focused on the likelihood of a recurrence of dumping should the anti-dumping measures be repealed. | Furthermore, the purpose of the imposition of the anti-dumping measures is not to stop the imports but to restore the level playing field on the Union market. |
| 6 | The investigation will be concluded, pursuant to Article 11(5) of the basic anti-dumping Regulation, within nine months of the date of the entry into force of this Regulation. | Nevertheless, it is recalled that the purpose of the imposition of the anti-dumping measures is not to stop the imports but to restore the level playing field on the Union market. |
| 7 | The recovery of the Union industry from past dumping practices was thus ongoing when the present investigation started. | Under the consistent case-law of the Court of Justice, the sole purpose of a regulation extending an anti-dumping duty is to ensure the effectiveness of that duty and to prevent its circumvention. |
| 8 | Under the consistent case-law of the Court of Justice, the sole purpose of a regulation extending an anti-dumping duty is to ensure the effectiveness of that duty and to prevent its circumvention. | The investigation will be concluded, pursuant to Article 11(5) of the basic anti-dumping Regulation, within nine months of the date of the entry into force of this Regulation. |
| 9 | The original anti-dumping investigation was initiated in February 2019 and definitive anti-dumping duties were imposed in April 2020 (see recital (1)). | According to WTO Panel report on United States – Anti-dumping and countervailing duties on certain products and the use of facts available (16), when applying facts available investigating authorities are required to select, in an unbiased and objective manner, those facts available that constitute reasonable replacements for the missing ‘necessary’ information in the specific facts and circumstances of a given case. |
| 10 | Their analysis allows the assessment of any undue negative impact on the parties concerned by the anti-dumping measures in place. | The allegations in the complaint requesting the initiation of an anti-dumping investigation estimate an average dumping margin of 123 % and an average injury elimination level of 43 % for the product concerned. |

Table 5: Top 10 answers to the query *”What is the purpose of an anti-dumping investigation?”*. Results between the pre-trained and fine-tuned models are different, and it seems the combined results from the fine-tuned model provide more information relevant to the query overall. We note that while the Spearman rank correlation between the gold labels and the model scores is of 0.16 for the pre-trained and 0.85 for the fine-tuned model on this query, the real improvement is mild.

5 Conclusion and Perspectives

5.1 Key takeaways

We’ve successfully addressed our objectives stated in sections 1.2 and 3.1 and summarize the key takeaways below:

- Ordinal labels can be a determining factor when drawing conclusions about different IR approaches.
- Cross-Encoder re-ranking consistently improves semantic search results on legal documents.
- Using BM25 in a scalable manner (with cross-encoder re-ranking of only the top 15 results) and without pre-processing does not yield satisfactory results.
- GPL is a good domain adaptation method, considering it is completely unsupervised and achieves results similar to some of the top performing pre-trained semantic search models.
- Augmented SBERT applied for semantic search requires many queries when trained with the Cosine Similarity Loss.
- Pre-trained semantic search sentence transformers appear to be the best approach to retrieve relevant passages on legal documents.

5.2 Drawbacks and Limitations

The main shortcomings of this study are that we’ve been unable to fine-tune better performing models than the pre-trained ones. This could be explained by the fact that the pre-trained models already work very well and the margin for improvement could be none-existent without any annotations. Regarding annotations, the labels we’ve gathered have both been used for evaluation purposes as well as fine-tuning with Augmented SBERT. About the former objective we’ve reported many information retrieval evaluation metrics and successfully compared the performance of our approaches using the suggested minimum of 50 queries (see section 3.2.2.1). We only evaluate the most relevant passages retrieved which still have been collected among all possible passages and can reasonably be used to compare different models.

As for the supervised fine-tuning objective, there is little to no documentation on the application of Augmented SBERT in a semantic search setting. On top of the apparent need for more queries, as suggested in section 4.3.4, the publication [33] also suggests that the silver dataset should match the distribution of our corpus, which is not necessarily our case, as it has only been constructed using the most relevant passages for multiple queries. While the augmentation most likely leads to highly irrelevant pairs, it fails to include other types of passages that have not been retrieved in the first place. Moreover, we’ve seen that judging a relevancy between 0 and 5 is a highly subjective task. Annotating triplets instead (is the first pair better than the second) might have been an easier task for our raters and might have improved the fine-tuning with a slightly different approach (using the MarginMSE or the MNR Loss for example - see section 2.4.3). Since the original Augmented SBERT used the Cosine Similarity Loss on a sentence pair regression

task, and since we also needed to use these labels for evaluation, we did not use triplet annotations.

5.3 Future work

Resulting from this project, we have obtained a large corpus of labeled query-passage pairs, which can be used for a variety of experiments. The entire dataset can be fine-tuned with supervised learning, but it should be evaluated on a new set of queries. As discussed in section 5.2, experimenting with different loss functions that are better suited for semantic search is another option. For example, if we use annotated triplets instead of pairs, we could apply the Augmented SBERT approach using the MarginMSE loss. This would resemble the GPL approach with an added step of manual validation of the generated query-answer pairs, which we believe to be a promising direction.

To expand the scope of our comparative study, different approaches based on late-interaction, lexical, sparse, or dense retrieval can be included. An in-depth analysis could also assess how complementary the pre-trained models are by examining the overlap between the retrieved passages of each model. Additionally, a multilingual IR study could be conducted by leveraging the numerous available translations of the initial documents and the annotations for the English ones. Further research ideas also include verifying whether our results align with a comparative study in a semantic similarity or clustering setting.

To conclude, it is worth noting that the AI powered chatbot ChatGPT¹⁹ was unveiled to the public during this project, generating a lot of buzz due to its impressive language processing capabilities. Our approach remains relevant in chat settings since generative models fail to provide the source of their data, and efficient and scalable retrieval remains an important initial stage to finding the sources on which to generate answers. Our methodology can serve as a robust baseline for future research in this field. With the power of pre-trained models and the potential for fine-tuning, the possibilities for further experimentation are vast.

¹⁹chat.openai.com

References

- [1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.
- [2] N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” in *EMNLP/IJCNLP (1)* (K. Inui, J. Jiang, V. Ng, and X. Wan, eds.), pp. 3980–3990, Association for Computational Linguistics, 2019.
- [3] K. Wang, N. Thakur, N. Reimers, and I. Gurevych, “GPL: Generative pseudo labeling for unsupervised domain adaptation of dense retrieval,” in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Seattle, United States), pp. 2345–2360, Association for Computational Linguistics, July 2022.
- [4] N. Thakur, N. Reimers, A. Ruckl’e, A. Srivastava, and I. Gurevych, “Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models,” *ArXiv*, vol. abs/2104.08663, 2021.
- [5] S. Robertson and H. Zaragoza, “The probabilistic relevance framework: BM25 and beyond,” *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.
- [6] R. Nogueira, W. Yang, J. Lin, and K. Cho, “Document expansion by query prediction,” *CoRR*, vol. abs/1904.08375, 2019.
- [7] Z. Dai and J. Callan, “Context-aware sentence/passage term importance estimation for first stage retrieval,” *CoRR*, vol. abs/1910.10687, 2019.
- [8] O. Khattab and M. Zaharia, “Colbert: Efficient and effective passage search via contextualized late interaction over bert,” in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’20, (New York, NY, USA), p. 39–48, Association for Computing Machinery, 2020.
- [9] M. Li and J. J. Lin, “Encoder adaptation of dense passage retrieval for open-domain question answering,” *ArXiv*, vol. abs/2110.01599, 2021.
- [10] S. Hofstätter, S.-C. Lin, J.-H. Yang, J. Lin, and A. Hanbury, “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*, SIGIR ’21, (New York, NY, USA), p. 113–122, Association for Computing Machinery, 2021.
- [11] J. Ma, I. Korotkov, Y. Yang, K. Hall, and R. McDonald, “Zero-shot neural passage retrieval via domain-targeted synthetic question generation,” in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, (Online), pp. 1075–1088, Association for Computational Linguistics, Apr. 2021.

- [12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.
- [13] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving language understanding by generative pre-training,” 2018.
- [14] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [16] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1724–1734, Association for Computational Linguistics, Oct. 2014.
- [17] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” 2019.
- [18] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, “Electra: Pre-training text encoders as discriminators rather than generators,” in *International Conference on Learning Representations*, 2020.
- [19] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, “Mpnet: Masked and permuted pre-training for language understanding,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS’20*, (Red Hook, NY, USA), Curran Associates Inc., 2020.
- [20] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” NIPS’20, (Red Hook, NY, USA), Curran Associates Inc., 2020.
- [21] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter,” *arXiv preprint arXiv:1910.01108*, 2019.
- [22] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, “Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS’20*, (Red Hook, NY, USA), Curran Associates Inc., 2020.

- [23] N. T.-H. Nguyen, P. P.-D. Ha, L. T. Nguyen, K. Van Nguyen, and N. L.-T. Nguyen, “Spbertqa: A two-stage question answering system based on sentence transformers for medical texts,” in *Knowledge Science, Engineering and Management: 15th International Conference, KSEM 2022, Singapore, August 6–8, 2022, Proceedings, Part II*, (Berlin, Heidelberg), p. 371–382, Springer-Verlag, 2022.
- [24] M. Grootendorst, “Bertopic: Neural topic modeling with a class-based tf-idf procedure,” 2022.
- [25] S. Humeau, K. Shuster, M.-A. Lachaux, and J. Weston, “Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring,” 2019.
- [26] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *EMNLP*, vol. 14, pp. 1532–1543, 2014.
- [27] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, “A large annotated corpus for learning natural language inference,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (Lisbon, Portugal), pp. 632–642, Association for Computational Linguistics, Sept. 2015.
- [28] D. M. Cer, M. T. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, “Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation,” in *SemEval@ACL* (S. Bethard, M. Carpuat, M. Apidianaki, S. M. Mohammad, D. M. Cer, and D. Jurgens, eds.), pp. 1–14, Association for Computational Linguistics, 2017.
- [29] M. L. Henderson, R. Al-Rfou, B. Strope, Y.-H. Sung, L. Lukács, R. Guo, S. Kumar, B. Miklos, and R. Kurzweil, “Efficient natural language response suggestion for smart reply,” *CoRR*, vol. abs/1705.00652, 2017.
- [30] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, and A. Hanbury, “Improving efficient neural ranking models with cross-architecture knowledge distillation,” *CoRR*, vol. abs/2010.02666, 2020.
- [31] K. Wang, N. Reimers, and I. Gurevych, “TSDAE: Using transformer-based sequential denoising auto-encoder for unsupervised sentence embedding learning,” in *Findings of the Association for Computational Linguistics: EMNLP 2021*, (Punta Cana, Dominican Republic), pp. 671–688, Association for Computational Linguistics, Nov. 2021.
- [32] C. Raffel, N. M. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *ArXiv*, vol. abs/1910.10683, 2019.
- [33] N. Thakur, N. Reimers, J. Daxenberger, and I. Gurevych, “Augmented SBERT: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Online), pp. 296–310, Association for Computational Linguistics, June 2021.

- [34] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [35] A. Trotman, A. Puurula, and B. Burgess, “Improvements to bm25 and language models examined,” in *Proceedings of the 2014 Australasian Document Computing Symposium*, ADCS ’14, (New York, NY, USA), p. 58–65, Association for Computing Machinery, 2014.
- [36] P. Yang, H. Fang, and J. Lin, “Anserini: Enabling the use of lucene for information retrieval research,” in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR ’17, (New York, NY, USA), p. 1253–1256, Association for Computing Machinery, 2017.
- [37] K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, and M. Zaharia, “ColBERTv2: Effective and efficient retrieval via lightweight late interaction,” in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Seattle, United States), pp. 3715–3734, Association for Computational Linguistics, July 2022.
- [38] N. Reimers, (*Twitter*) “GPT-3 embeddings by @openai was announced this week. I was excited and tested them on 20 datasets. sadly they are worse than open models that are 1000 x smaller. running @OpenAI models can be a 1 million times more expensive”. Jan 2022.
- [39] C. D. Manning, P. Raghavan, and S. Hinrich, *Evaluation in information retrieval*. Cambridge University Press, 2019.
- [40] L. McInnes, J. Healy, and J. Melville, “Umap: Uniform manifold approximation and projection for dimension reduction,” 2018.
- [41] C. Malzer and M. Baum, “A hybrid approach to hierarchical density-based cluster selection,” 2020.
- [42] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, “Ms marco: A human generated machine reading comprehension dataset.,” *CoRR*, vol. abs/1611.09268, 2016.
- [43] M. Honnibal and M. Johnson, “An improved non-monotonic transition system for dependency parsing,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, (Lisbon, Portugal), pp. 1373–1378, Association for Computational Linguistics, Sept. 2015.
- [44] J. Nivre and J. Nilsson, “Pseudo-projective dependency parsing.,” in *ACL* (K. Knight, H. T. Ng, and K. Oflazer, eds.), pp. 99–106, The Association for Computer Linguistics, 2005.
- [45] M. Honnibal and I. Montani, “spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.” 2017.
- [46] S. Wu and S. I. McClean, “Information retrieval evaluation with partial relevance judgment.,” in *BNCOD* (D. A. Bell and J. Hong, eds.), vol. 4042 of *Lecture Notes in Computer Science*, pp. 86–93, Springer, 2006.

- [47] S. Teufel, “An overview of evaluation methods in trec ad hoc information retrieval and trec question answering,” in *Evaluation of Text and Speech Systems*, p. 163–186, Springer, 2007.
- [48] C. Buckley and E. M. Voorhees, “Evaluating evaluation measure stability,” in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '00, (New York, NY, USA), pp. 33–40, ACM, 2000.

Appendix

A Queries

What are the duties of eu reference laboratories?
What does the enterprise europe network do?
What kind of securities are eligible for eu guarantee?
Which decision clarifies the provision of a network code?
What are underlying exposures?
What is EUROSUR?
When did the european border surveillance system begin?
What are the general conditions for reusing data?
Freedoms provided by the eea agreement?
What are strategic investments?
What is the esf+?
How do esf+ contribute to gender equality?
What is a dwelling?
What does anonymous mean in the context of data?
How is personal data reused?
Can biodiesel be exported?
Do credit purchasers need a license?
Do members of the eu need to develop a plan for security?
Does the eu issue its own certificate of authenticity?
How are company incomes calculated?
How does the European Union measure market value?
How long is an identifying document to be kept?
How long will eu retain a customer's personal data?
How much can be caught with a fishing authorisation?
How to change the fishing authorisation?
Regulated medicines definition
What are the basic safety requirements for stationary bicycles?
What are the macroeconomic indicators?
What are the main requirements of the energy security directive?
What are the principles for reusing data?
What does an accredited verifier do?
What does een mean?
What does european security mean?
What does self employment mean?
What investments are covered under the eu guarantee?
What is a classified production area?
What is a commercial invoice?
What is a proximate chain of events?
What is a reference laboratory?
Which eu institution is involved in clearing accounts?
Elisa test definition
Eurojust definition
What are investment guidelines?
What is the csd in a financial statement?
What is the maximum spectrum level allowed for rlan?
What is the purpose of an anti-dumping investigation?
What is the purpose of the oppf tax?
What method is used to test tomatoes?
Which state is involved in switzerland's compliance protocol?
What type of inspection do you require for a sanitary survey?
What do u need to do for biodiversity?

Table 6: (Q_2)

What are the requirements for a livestock license?
 When do railway licences expire?
 What is a low capacity slaughterhouse?
 What are liquidity requirements for investment firms?
 What date does the european monopoly act start?
 What is an ecolabel?
 What is considered a reasonable profit?
 How do syndicated loans work?
 What does the european agreement on scientific cooperation include?
 What are the requirements for a crowdfunding service provider?
 What is a life cycle assessment?
 What does selective distribution mean?
 How long can an eu blue card be used?
 What is cypermethrin used for?
 What is the name of the italian company from the european union that collects protected designations of origin and protected geographical indications?
 EU regulation to protect personal data
 What is the definition of an appeal?
 Define covered bonds
 How to transfer personal data internationally?
 What is a free trade zone?
 What is a vms?
 What measures should hosting service providers take to prevent terrorism?
 What is the purpose of data collection?
 What was the purpose of the conference on chemical weapons?
 Who is responsible for calculating the emissions of a car?
 What is the definition of a carbon leakage?
 Types of petroleum products
 What is covid-19?
 What is eu-LISA?
 What is the e-codex?
 What causes the decrease in natural gas prices?
 Recommended daily dose for milking?
 What is a public key infrastructure?
 What is considered a sampling scheme?
 What is the method for testing genetically modified food?
 Who can propose amendments to rules of procedure?
 How often should carbon emissions be measured?
 What is the necessary step of the process to protect designation of origin?
 What is the cn code for maize?
 What is a zero emissions vehicle?
 What is a hedging instrument?
 What is an information system?
 How can antidumping duties be effective?
 What type of goods do EU imports include?
 When is an item of information considered confidential?
 Who is mayor of cugir city?
 Where is rum manufactured?
 What is an environmental performance indicator for public administration?
 What regulation is applicable for metadata?

Table 7: (Q_3)

B BERTopic



Figure 21: First 32 Topic c-TF-IDF Representations for the EUR-Lex data. These are the 32 topics containing most sentences and obtained with BERTopic's default parameters for embeddings selection, UMAP, HDBSCAN.

C Inter-Annotator Agreement

The data introduced in section 4.1 is presented again in Figure 22, this time in heatmap form. The cell with most annotations corresponds to a score of 3 given by annotator A considered as a score of 4 by annotator B. This shift is not unidirectional though, as we can see many pairs annotated as 5 by annotator A have also been considered as 4 by annotator B. While such differences are acceptable, answers with strong disagreements require further inspection. Most often, pairs with disagreements are not easy to annotate at first sight and have either been annotated too quickly by one annotator or have been annotated with some external knowledge. An example of extreme disagreement is shown in Figure 23.

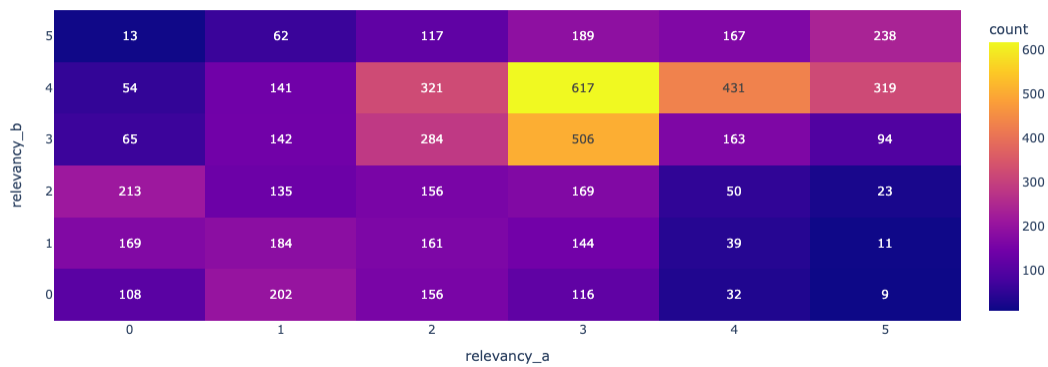


Figure 22: Relevancy Heatmap: Annotator A vs. Annotator B. Using the data from section 4.1.

Commission Implementing Regulation (EU) 2018/1287 of 24 September 2018 granting a Union authorisation for the biocidal product family Quat-Chem's iodine based products (Text with EEA relevance.)

HDPE CONTAINER 1 000 l 4.1.1. Where specific to the use, the particulars of likely direct or indirect effects, first aid instructions and emergency measures to protect the environment See general directions for use of meta SPC 4 4.1.4. Where specific to the use, the instructions for safe disposal of the product and its packaging None: see directions for use meta SPC 4 4.1.5. Where specific to the use, the conditions of storage and shelf-life of the product under normal conditions of storage See general directions for use of meta SPC 4 5.

— cows and buffaloes (3 to 10 ml: 5 ml recommended) — sheep (1,5 to 5 ml: 1,5 ml recommended) — goats (2,5 to 6 ml: 2,5 ml recommended) Leave the product until next milking.

At the next milking, use the teat cleaning and wiping method systematically before attaching the milking cluster. In case a combination of pre- and post-milking disinfection is necessary, using another biocidal product not containing iodine has to be considered for pre-milking disinfection. Particulars of likely direct or indirect effects, first aid instructions and emergency measures to protect the environment Take the contaminated clothes and shoes off immediately. Instructions for safe disposal of the product and its packaging At the end of

Does it answer the question : recommended daily dose for milking ?

0 1 2 3 4 5

Figure 23: Example of a query-passage pair where one rater considered maximum relevancy 5 and the other rater considered relevancy score 0. At first glance it seems like a perfect answer but the context combined with some external knowledge indicates the quantities refer to milk certification protocols rather than a recommended daily dose.

If we isolate the query-passage pairs that have been removed from our results and plot the number of strong disagreements versus the length of their associated queries (see Figure 24), we see a moderate correlation. The correlation coefficient is in fact 0.45. As suggested in section 3.2.2.2, longer queries lead to results that are more difficult to retrieve and evaluate, which is partially verified here.

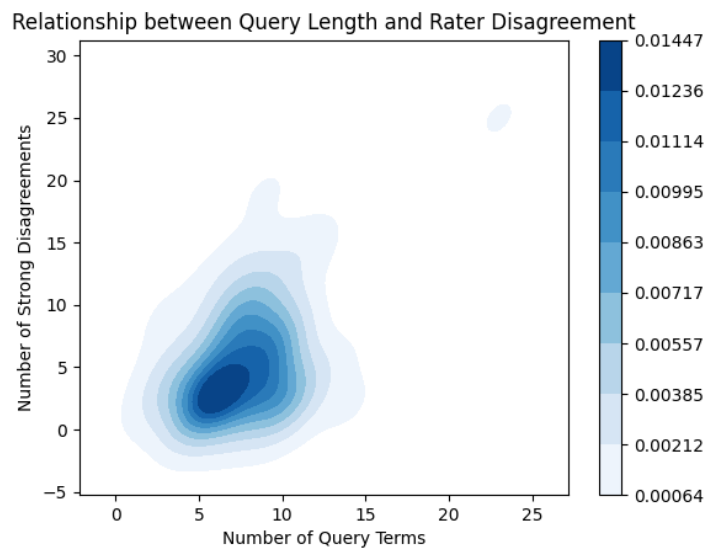


Figure 24: Kernel Density Estimate: Number of Query Terms vs. Number of Strong Disagreements. Using the data from section 4.1 where we've kept only the pairs with strong disagreements (relevancy score difference ≥ 3).