Heidelberg University
Institute of Computer Science
Summer Term 2023
Seminar: Modern Information Retrieval
Instructors: Prof. Dr. Michael Gertz
Nicolas Reuter
Ashish Chouhan
Jayson Salazar
John Ziegler
Satya Almasian

Seminar Paper

Dense Retrieval with Entity Views

Name:	Johannes Gabriel Sindlinger
Student ID:	3729339
Study Programme:	Master, Computer and Data Science (3rd semester)
Email:	johannes.sindlinger@stud.uni-heidelberg.de
Date of Submission:	10/08/2023

Hiermit versichere ich, **Johannes Gabriel Sindlinger**, dass ich die Hausarbeit mit dem Titel **Dense Retrieval with Entity Views** im Seminar **Modern Information Retrieval** im **Sommersemester 2023** bei **Prof. Dr. Michael Gertz** und **Nicolas Reuter**, **Ashish Chouhan**, **Jayson Salazar**, **John Ziegler** und **Satya Almasian** selbstständig und nur mit den in der Arbeit angegebenen Hilfsmitteln verfasst habe. Zitate sowie der Gebrauch fremder Quellen, Texte und Hilfsmittel habe ich nach den Regeln wissenschaftlicher Praxis eindeutig gekennzeichnet. Mir ist bewusst, dass ich fremde Texte und Textpassagen nicht als meine eigenen ausgeben darf und dass ein Verstoß gegen diese Grundregel des wissenschaftlichen Arbeitens als Täuschungs- und Betrugsversuch gilt, der entsprechende Konsequenzen nach sich zieht. Diese bestehen in der Bewertung der Prüfungsleistung mit „nicht ausreichend“ (5,0) sowie ggf. weiteren Maßnahmen.

Außerdem bestätige ich, dass diese Arbeit in gleicher oder ähnlicher Form noch in keinem anderen Seminar vorgelegt wurde.

Heidelberg, 10.08.2023 ________________

Contents

1 Introduction

In the field of information retrieval, finding relevant information efficiently and effectively from large collections has been a challenge for a long time. This task involves finding documents or passages that are most relevant to a particular query or topic. Conventional methods of information retrieval such as BM-25 (Robertson and Zaragoza [19]) often rely on sparse representations of documents, that miss subtle semantic connections and fail to fully capture the context of the information being searched.

To address the limitations of sparse retrieval approach, dense retrieval has emerged as an alternative approach. This approach entails the utilization of dense vector representations for documents and queries, signifying continuous-valued vectors situated within a high-dimensional space, commonly referred to as embeddings. In the context of dense retrieval these are mostly built using pre-trained language models such as BERT (Devlin et al. [5]) to encode rich semantic information and enabling comprehensive understanding of content.

Two main approaches to dense retrieval are the bi-encoder and cross-encoder methods. The bi-encoder approach involves creating distinct embeddings for queries and documents, which are then compared by similarity measurements as cosine similarity. On the other hand, the cross-encoder approach directly measures the similarity between the query and document by taking both as input for its pre-trained language model. This approach results in increased computational expenses as a consequence of performing model inference for each pair of query and document.

Recently, information retrieval has evolved from solely ranking matching documents to a 'rich search' or 'Entity-oriented search' (Balog [2]). 'Rich search' implies that queries often require specific information about entities, facts, or structured data for more sophisticated tasks.

However, pre-trained language models like BERT suffer from an issue within this context: entities such as people, places, or organizations are not entirely represented in the models (Heinzerling and Inui [8]). To substantiate this assertion, Heinzerling and Inui refer to the training process of BERT model, which utilizes the well-known masked language model approach. During training phase, the model is demanded to predict randomly masked tokens based on the context of the surrounding words. In the example, 'Paris is the capital of [MASK]' (Heinzerling and Inui [8]), 'France' is correctly predicted with a high probability, demonstrating the model's ability to recognize frequent entities. However, prediction might be ambiguous for rarer entities like 'Sesame Street' in the case of 'Bert is a character on [MASK]' (Heinzerling and Inui [8]). In particular, entities that were not present during the training process and only emerged afterwards cannot be captured within pre-trained language models.

This limitation motivated Tran and Yates [22] to explore solutions in their work 'Dense Retrieval with Entity Views', which is the subject of this seminar report. The subsequent sections will present and critically examine Tran and Yates' work. First, I will contextualize the paper in current research in section 2, explain the methods used in section 3, present the results in section 4, and conclude with a critical assessment and personal opinion in section 5.

2 Related Work

Entities have been a subject of interest in the field of information retrieval since the emergence of knowledge repositories like Wikipedia. Research in this context can be divided into the time before and after the advent of pre-trained language models such as BERT.

2.1 Entities in Sparse Retrieval

Prior to the advent of pre-trained language models such as BERT, research focused on sparse retrieval methods like BM-25, where entities from knowledge bases were used to enhance query understanding. Recent work in this context focuses mainly on extending queries with additional data: Xu et al. [26] proposed a method to expand queries by Wikipedia descriptions of entities. Other researchers like Dalton et al. [4] extended this idea and extracted more sophisticated features out of the linked knowledge bases like synonyms or relationships to other entities. The research in this field is quite extensive and has been summarized in detail in a meta-work by Balog [2].

2.2 Entities in Dense Retrieval

While some research has already addressed the consideration of entities in sparse retrieval, the amount of work in the field of dense retrieval in this context is rather limited.

Some work focuses on integration of entities within interaction-based methods: An example of this approach is the work of Xiong et al. [23, 25]. They developed a method to build ranking features by incorporating an attention mechanism of word embeddings and entity embeddings and significantly outperformed baselines for word-based and entity-based learning to rank systems. Interaction-based methods can be considered as extensions of the cross-encoder approach of dense retrieval and therefore face the issue of high computational complexity as described in section 1. All documents and query pairs must be inferred, which can lead to significant time and resource overhead. This leads to a slow overall retrieval process and can be inefficient, especially with large data sets.

In the context of the broad field of natural language processing tasks, entities have been considered in several research papers and shown to have a significant impact on its tasks (e.g. Liu et al. [12], Peters et al. [17]). ERNIE model proposed by Sun et al. [21] incorporates knowledge from both pre-training tasks and external knowledge graphs, enabling it to achieve better contextual understanding and knowledge integration in natural language processing tasks.

The existing concepts of dense retrieval, including interaction-based methods and models like ERNIE, do not fully explore entities independently from the underlying pre-trained language model for the traditional retrieval task. This represents the main novelty of Tran and Yates' work [22], which will be elaborated in the following sections.

3 Methodology

Tran and Yates' [22] main contribution is to consider entities embeddings independently of the document or query embeddings obtained from pre-trained language model. To achieve this, they combine embeddings from pre-trained language model TAS BERT (Hofstätter et al. [9]) with Wikipedia2Vec (Yamada et. al [27]) embeddings of entities extracted from documents and queries using. When combining with the document embeddings, the embeddings of entities provide multiple views on the same information, with different sets of entities representing various perspectives on the queries and documents. Depending on the displayed view, different relevant documents or sections of documents can be identified.

Tran and Yates do not use the information about documents and entities as an input for a learned framework, deviating from the cross-encoder approach (see section 1). Instead, they merge these embeddings into a joint vector space. This joint vector space results in a final embedding used to calculate similarities between vectors. Thus, Tran and Yates adopt the bi-encoder model (as discussed in section 1), allowing for indexing of document embeddings and enabling fast ranking computations through ANN search.

3.1 General Model

Tran and Yates' proposed method involves independent embeddings of documents and entities. The process entails merging the embeddings of both documents and entities for each query and document to achieve a joint vector space, as depicted in Figure 1. Following the bi-encoder approach, the embeddings for queries and documents are created independently using the pre-trained language model TAS BERT (Hofstätter et al. [9]), which builds upon a distilled version of BERT (Sanh et al. [20]).

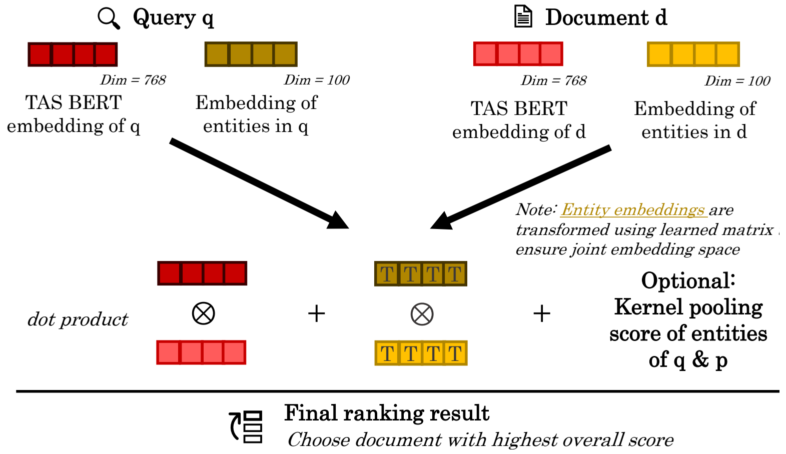

Query q
Dim = 768
TAS BERT embedding of q
Dim = 100
Embedding of entities in q
Document d
Dim = 768
TAS BERT embedding of d
Dim = 100
Embedding of entities in d
Note: Entity embeddings are transformed using learned matrix
ensure joint embedding space
T T T T
dot product
⊗
+
⊗
T T T T
+
Optional: Kernel pooling score of entities of q & p
Final ranking result
Choose document with highest overall score

Figure 1: General model

The TAS BERT approach employs topic-aware sampling for fine-tuning on information retrieval task, thus queries in the training dataset are clustered based on different topics. During training, query samples are extracted exclusively from clusters with identical topics, aiming to enhance the sensitivity of the information retrieval model to different topics.

In the dense retrieval setting, embeddings of tokens are compared using similarity measurements to rank the best documents concerning a query. Tran and Yates follow a similar principle but enhance it by incorporating embeddings of entities. For each query and document, a single entity embedding is generated and concatenated with the textual embedding to create an embedding that captures both the semantic context and entity information. Final ranking is built upon dot product as similarity measurement.

Experimental results demonstrate that concatenation produces superior results for combining text and entity embeddings compared to other methods as max pooling and sum pooling. Table 1 displays the respective analysis of Tran and Yates based on their experimental setup, which is introduced in detail in section 4. Apart from the analytical point of view, the approach of concatenation offers the advantage that it can be understood quite intuitively.

Operators	**MS Marco**		
	nDCG	**MRR**	**MAP**
Sum	0.393	0.335	0.339
Max	0.388	0.330	0.334
Concat	0.396	0.341	0.343

Table 1: Varying Aggregation Operators of Embedding Concatenation

However, concatenation introduces a challenge as the word embeddings and entity embeddings are generated in different vector spaces with varying dimensions and magnitudes. To address this issue, Tran and Yates introduce a transformation matrix $W \in \mathbb{R}^{100\times 100}$ to transform the entity embeddings into the joint vector space. So let $\mathbf{E}(t) \in \mathbb{R}^{100}$ be the embedding of entities of text t, the transformed entity embedding is given as:

$$\mathbf{R}_{entity}(t) = W^T \cdot \mathbf{E}(t) \tag{1}$$

The values of the matrix are determined during training process and thus reflect a meaningful transformation of the embeddings into a joint vector space.

Given a text t with its corresponding word embedding $\mathbf{R}_{text}(t)$ derived from the pre-trained language model, the final vector representation of t is obtained by concatenating the transformed entity embedding $\mathbf{R}_{entity}(t)$ and the word embedding:

$$\mathbf{R}_{final}(t) = \mathbf{R}_{text}(t) \oplus \mathbf{R}_{entity}(t) \tag{2}$$

The ranking score value of a pair of query q and document d is then obtained using the dot product $\otimes$ as a similarity measure:

$$\begin{aligned} \mathbf{Score}(q, d) &= (\mathbf{R}_{final}(q) \otimes \mathbf{R}_{final}(d)) && (3) \\ &= (\mathbf{R}_{text}(q) \otimes \mathbf{R}_{text}(d)) \oplus (\mathbf{R}_{entity}(q) \otimes \mathbf{R}_{entity}(d)) && (4) \end{aligned}$$

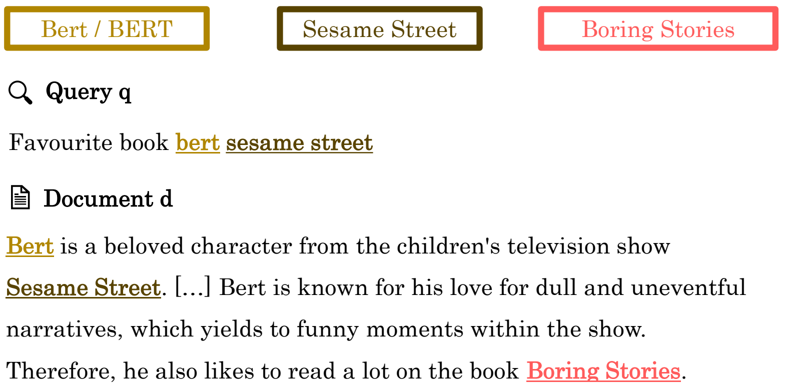
As an optional addition to their model, Tran and Yates include a kernel based neural ranking model (KNRM, Xiong et al. [24]) as an external scoring source, which predicts the similarity between document and query entities. This interaction-based approach requires knowledge of the query and document at runtime. For details, refer section 3.3. Incorporating the KNRM signal S_{knrm} into the final ranking score yields the score with KNRM signal, denoted by $\mathbf{Score}_{knrm}(q, d)$:

$$\mathbf{Score}_{knrm}(q, d) = (\mathbf{R}_{final}(q) \otimes \mathbf{R}_{final}(d)) + S_{knrm} \tag{5}$$

3.2 Generating Entity Embeddings

As described in section 3.1, Tran and Yates create embeddings for both text and entities in queries and documents. While the text embeddings are generated using TAS BERT, the primary focus of the contribution of Tran and Yates lies in the creation of entity embeddings. In particular, as queries and documents often contain multiple entities, these entities need to be aggregated into a single embedding.

To illustrate this process, consider the example in Figure 2. In this example, the query 'Favourite book bert sesame street' corresponds to a document that mentions three entities: Bert, Sesame Street and Boring Stories.



Favourite book bert sesame street

Bert is a beloved character from the children's television show Sesame Street. [...] Bert is known for his love for dull and uneventful narratives, which yields to funny moments within the show. Therefore, he also likes to read a lot on the book Boring Stories.

Figure 2: Example query and example document

3.2.1 Extracting Entities

To aggregate multiple entities in a document or query, Tran and Yates first extract these entities from the text using external frameworks Dexter (Ceccarelli et al. [3]) and Wikipedia2Vec (Yamada et. al [27]).

- The Dexter framework is an entity linkage system that resolves entity mentions in text to corresponding entities in a knowledge base. In particular, Dexter employs a combination of methods, including named entity recognition and pattern matching, to perform this task.
- Wikipedia2Vec leverages the structure of Wikipedia to generate dense vector representations (embeddings) for words, articles, and entities. It utilizes the Word2Vec algorithm (Mikolov et al. [15]) to capture semantic relationships from Wikipedia, producing high-dimensional embeddings.

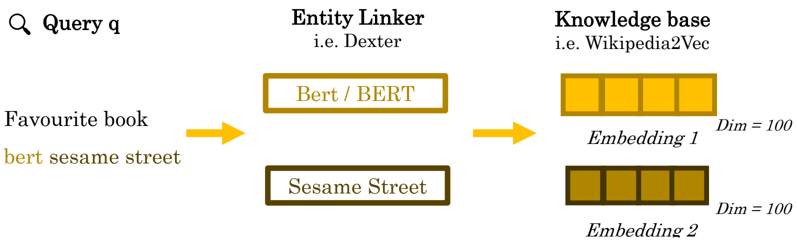


Figure 3: Process of entity extraction

The extraction procedure involves submitting a document or query to Dexter, which extracts entity mentions from the given text. These entity names are then passed on to the knowledge base, which transforms them into embeddings. Wikipedia2Vec provides vectors in dimension 100 as default. Tran and Yates keep this value and therefore generate entity embeddings in dimension 100. Figure 3 visualizes this process.

3.2.2 Combining Entities for Queries

For now, multiple embeddings for each entity within a query or a document are generated by applying the procedure outlined in section 3.2.1. The aggregation of these generated embeddings differs based on whether a query or a document is considered, given the usual brevity of queries compared to documents.

For queries, Tran and Yates adopt a straightforward approach. They aggregate the embeddings of entities by averaging all entity embeddings across all dimensions. Figure 4 visualizes this procedure.

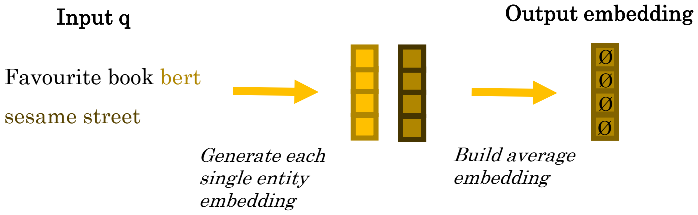


Figure 4: Process of generating a single entity embedding for a query

3.2.3 Combining Entities for Documents

Documents usually contain many different entities, as the example in Figure 2 indicates. Additionally, documents might also cover different aspects of a topic, so entities from very different areas might appear within them. To handle this additional complexity, the authors proposed three successive methods for entity aggregation in documents, namely:

- Single Entity Representation (EVA Single)
- Query-Aware Single Entity Representation (EVA Single-QA)
- Multiple Entity View Representation (EVA Multi)

The term 'EVA' stands for Entity Views in Dense Retrieval. The concept of 'Entity Views' which gives the paper its name, is particularly relevant in the third method, EVA Multi, and will be elaborated in the following sections.

EVA Single The initial approach for aggregating entities in a document, called EVA Single, is similar to the one used for queries. The key idea behind the Single Entity Representation is to extract all entities present in the document and subsequently generate a single output embedding by computing the average of these entity embeddings. This process is analogous to the one depicted in Figure 4 for queries, and a visual representation for documents is illustrated in Figure 5.

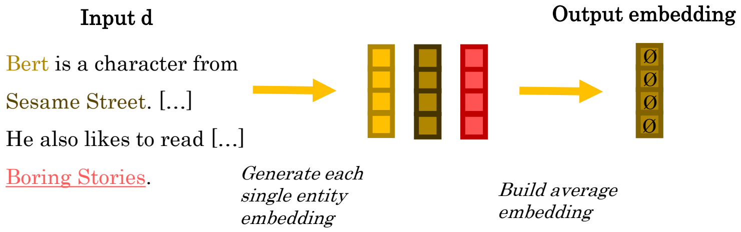


Figure 5: Process of generating an entity embedding for a document following the EVA Single approach

Despite its simplicity, this approach overlooks the fact that documents may cover diverse topics. By discarding query information, the EVA Single approach considers all entities, including those that may be only partially relevant to the document's main topic. Consequently, this disregards the relevance of entities within the ranking process, leading to biased results, as observed in section 4.

EVA Single-QA To address the problem of EVA Single disregarding the entities of queries, Tran and Yates introduce the Query-Aware Single Entity Representation. This approach creates embeddings that are tailored to the specific needs of a given query. However, this improvement comes with increased computational complexity since it assumes knowledge of the query beforehand. As a result, calculations for all query-document pairs must be performed during runtime, eliminating the possibility of precomputing document embeddings and indexing, leading to higher query latency, as observed in section 4.

The underlying idea of the EVA Single-QA model is to filter the entities of a document based on the information of a given query and select only entities with high similarity to a query entity. Figure 6 visualizes this process.

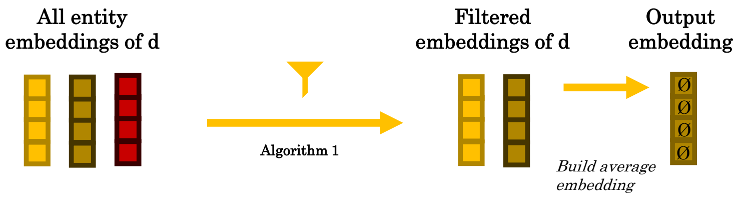


Figure 6: Process of generating an entity embedding for a document following the EVA Single-QA approach

The filtering is conducted using Algorithm 1, which is applied to each query-document pair (q, d). For every entity within q, the algorithm searches for the corresponding entity in d that possesses the maximum cosine similarity. If the similarity exceeds a specified threshold α, the respective entity in d is added to the filtered list $X_{focus}(d)$. Subsequently, the single output embedding for document d is computed as the average of all entity embeddings within $X_{focus}(d)$.

Algorithm 1 Query-aware document entity representation

```
Input: Query $q$ and document $d$, threshold $\alpha$
Output: Filtered entity embedding list $X_{focus}(d)$ of $d$
$X(q) \leftarrow$ set of embeddings of entities in $q$
$X_{focus}(d) \leftarrow \{\}$
for $e$ in $X(q)$ do
    $e^* \leftarrow$ entity embedding in $d$ having the maximum cosine similarity with $e$
    if cosine similarity$(e^*, e) > \alpha$ then
        $X_{focus}(d) \leftarrow X_{focus}(d) \cup \{e^*\}$
    end if
end for
return $X_{focus}(d)$
```

EVA Multi To address the issue of known queries required by the EVA Single-QA approach, Tran and Yates propose the EVA Multi method, which introduces multiple entity views as a solution with minimal negative consequences.

Analysis of training data revealed that in the vast majority of instances, the number of entities in queries does not exceed two. As shown in Table 2, in the MS MARCO dataset, 99.6 % of the 300,000 training instances and 99.5 % of the test instances contain two or fewer entities. Based on this observation, Tran and Yates focused their research on the assumption that it is sufficient to consider a maximum of two entities in queries.

Entities	**Training Queries**		**Testing Queries**	
	Count	**Fraction**	**Count**	**Fraction**
0	130,353	0.435	3,442	0.483
1	149,073	0.497	3,232	0.454
2	19,207	0.064	416	0.058
3+	1,367	0.004	37	0.005
Total	300,000		7,127	
Average	0.640		0.587	

Table 2: Summary statistics of queries

Under this assumption, when applying Algorithm 1, at most two entities remain in the filtered embedding list X_{focus}. This limitation stems from the algorithm iterating over the set of all entities in the given query once, resulting in X_{focus} containing either zero, one, or at most two items. Therefore, the number of all possible sets eligible for X_{focus} is bounded.

Leveraging this observation, Tran and Yates introduce clusters of entities, which represent different views of a document. In the EVA Multi approach, all possible single itemsets and sets of pairs of the entities are generated, yielding set *clusters*. However, sets of pairs are considered only if the cosine similarity between the two entities within a pair exceeds a predefined threshold β, ensuring only relevant entity views are generated. Subsequent algorithm 2 carries out this procedure.

Algorithm 2 Multiple cluster sets of document

```
Input: Document $d$, maximum cluster size $M$ $(= 2)$
Output: Multiple cluster total representations of $d$
$X(d) \leftarrow$ set of all entities in $p$
$clusters \leftarrow \emptyset$
for every non-empty subset $C \subset X(d)$ with size $l \leq M$ do
    if $l = 1$ or (every pair of entities in $C$ has cosine similarity $> \beta$) then
        $clusters \leftarrow clusters \cup C$
    end if
end for
return $clusters$
```

The final output embeddings are then calculated by averaging the embeddings of all items within each set in *clusters*. When applying the optional KNRM signal (see section 3.3) to this approach, the entity interaction matrix T is built only upon the set of entity embeddings within a single cluster, rather than considering all entities within the document.

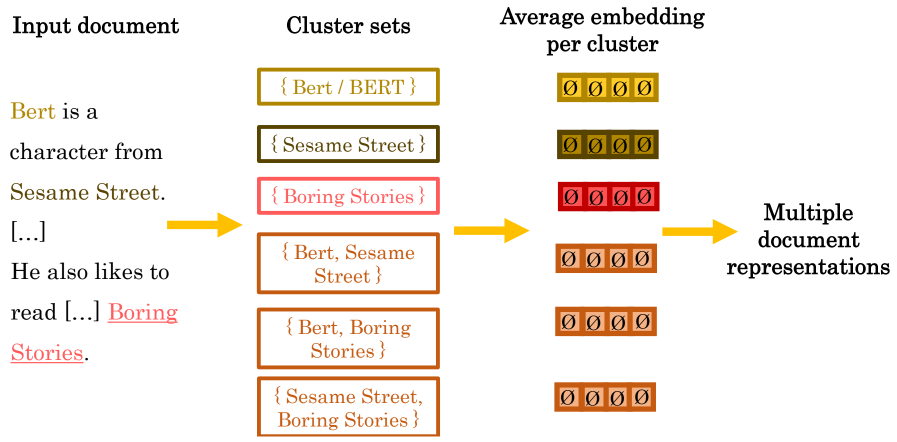

Input document
Cluster sets
Average embedding per cluster
Bert is a character from Sesame Street. [...] He also likes to read [...] Boring Stories.
{ Bert / BERT }
{ Sesame Street }
{ Boring Stories }
{ Bert, Sesame Street }
{ Bert, Boring Stories }
{ Sesame Street, Boring Stories }
Ø Ø Ø Ø
Multiple document representations

Figure 7: Process of generating entity embeddings for a document following the EVA Multi approach

This allows all calculations to be performed independently of query information, enabling indexing. Unlike the previous methods, EVA Single and EVA Single-QA, the EVA Multi approach generates several entity output embeddings per document. Figure 7 visualizes the process of generating the embeddings of multiple entity views or clusters.

3.3 KNRM Signal

Tran and Yates introduce a kernel based neural ranking model (KNRM), originally developed by Xiong et al. [24], as an additional scoring mechanism to extend their approach of incorporating entity embeddings into bi-encoder model. Unlike its original use, Tran and Yates adapt the KNRM model to specifically capture the interaction between entities in queries and documents. Equation 5 depicts the integration of the KNRM signal S_{knrm} into the calculations for the final ranking score, which Tran and Yates apply for the EVA Single-QA and EVA Multi models, yielding in EVA Single-QA-KNRM and EVA Multi KNRM models (see section 4).

The calculation of the KNRM signal follows these steps:

1. Let $X(q)$ be the set of all entity embeddings within query q, $X(d)$ any set of entity embeddings which occur in document d. For EVA Single and EVA Single-QA models (see section 3.2.3) $X(d)$ contains all entities with the given document, for EVA Multi (see section 3.2.3)

$X(d)$ contains only entities of specific subsets of entities of d. The entity interaction matrix is defined as:

$$T_{i,j} := sim(X_i(q), X_j(d)) \ ,$$

where $X_i(q)$ and $X_j(d)$ are the i-th and j-th entity embedding of q and d. The similarity function is given as cosine similarity. Embeddings are generated via Wikipedia2Vec (Yamada et. al [27]), as described in section 3.2.1.

2. Build k kernels using radial basis function, which creates differentiable histograms around hyperparameters μ_i and σ_i^2 for $i \in \{1, \dots, |X(q)|\}$.

$$K_l(X_i(q)) = \sum_{j=1}^{|X(p)|} \exp\left(-\frac{(T_{i,j} - \mu_i)^2}{2\sigma_i^2}\right)$$

3. Pool / Summarize the k results into a k-dimensional feature vector:

$$\overrightarrow{K(X_i(q))} = [K_1(X_i(q)), \dots, K_k(X_i(q))]$$

4. Build kernel-pooled representation $\phi(T)$ by calculating log-sum for each query entity:

$$\phi(T) = \sum_{i=1}^{|X(q)|} \log \overrightarrow{K(X_i(q))}$$

5. Get final kernel pooling score by applying a learned ranking layer. Note that $\tanh(\cdot) \in (-1, 1)$ and therefore $\sup S_{knrm} = 1$:

$$S_{\text{knrm}} = \tanh(w^T \phi(T) + b)$$

Despite being an interaction-based model that requires scoring during runtime for all query-document pairs, the computational complexity of the KNRM approach remains limited. The computations involved are relatively straightforward, and the additional learned layer does not significantly increase the computational overhead. Furthermore, these computations can be performed in parallel with the other components of Tran and Yates' approach. Empirical results concerning the latency of the models, with and without the KNRM signal, validate this assertion (see section 4).

4 Results

In this section, the concrete implementation of the described models in section 3 and the resulting results are presented. The focus of the elaborations is on the major findings of the work by Tran and Yates; accordingly, not all results are listed herein. Details can be studied in the paper by Tran and Yates [22].

4.1 Experimental Setup

Tran and Yates implemented their models in a TensorFlow (Abadi et al. [1]) setup: For the pretrained language model, they chose a distilled TAS BERT (Sanh et al. [20], Hofstätter et al. [9]), as described in section 3.1. To determine the embeddings, they used Dexter (Ceccarelli et al. [3]) and Wikipedia2Vec (Yamada et. al [27]), as explained in section 3.2.

Training was conducted using pairwise hinge loss on four Quadro RTX 8000 GPUs in parallel, employing 300,000 training samples from the MS MARCO dataset (Nguyen et al. [16]). For the models allowing indexing (EVA Single and EVA Multi, see section 3.2.3), the end-trained model was used to index embeddings for documents. Evaluation was performed on 7,127 test samples from the MS Marco dataset and the datasets TREC Deep Learning (DL) Track 2019 (MacAvaney et al. [13]), TREC DL 2020 (MacAvaney et al. [14]), TREC DL HARD (Yates et al. [28]). Evaluation metrics were chosen to be nDCG@10, MRR@10, MAP@1000.

The models proposed by Tran and Yates (EVA Single, EVA Single-QA, and EVA Multi, as described in section 3.2.3) were compared against various baselines as BM-25, TAS BERT, ERNIE and others. Details can be examined within the paper by Tran and Yates [22].

4.2 Efficiency

In terms of efficiency, particular attention is paid to latency, and the results for all models under examination are presented in Table 3. The table displays the average search time per query across evaluation datasets for each model, executed on the same server.

Methods	**Latency (ms)**
Low latency (<100 ms)	
BM25	13
ANCE	25
ERNIE Tuned	29
ERNIE Multi	70
TAS BERT	28
EVA Single	40
EVA Multi	76
EVA Multi-KNRM	74
Higher latency (>100 ms)	
EVA Single-QA	2,039
EVA Single-QA-KNRM	3,839
BM25 + T5 (Zero-Shot)	5,052
Best Reported	-

Table 3: Analysis of effectiveness of EVA models and baselines

4.3 Effectiveness

The effectiveness results of the various models applying nDCG@10, MRR@10, MAP@1000 consistently hold across all combinations of evaluation metrics and datasets. To avoid redundancy, this report will provide exemplary results based on the nDCG@10 metric and the TREC DL Track 2019 dataset [13]. These exemplary results are illustrated in a visual format in Figure 8. Further details on the main outcomes can be found in the subsequent section 4.4.

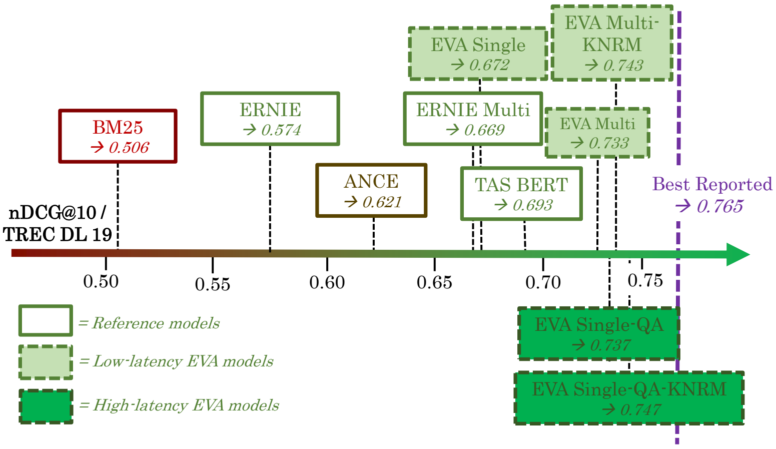


Figure 8: Exemplary results of EVA models and baselines for TREC DL 19 dataset and evaluation metric nDCG@10

4.4 Impact

The outcomes of Tran and Yates' work, based on the results from section 4.3 and section 4.2, is summarized as follows:

Enriching pre-trained language models with entity embeddings improve effectiveness significantly: The effectiveness results demonstrate that the models proposed by Tran and Yates, namely EVA Multi and EVA Single-QA, both with and without KNRM signal, outperform the various baselines. In the example presented in Figure 8, the nDCG@10 value for the best performing baseline, TAS BERT, is 0.693. In contrast, the values for EVA Multi and EVA Single-QA are 0.733 and 0.737. Moreover, the models only marginally deviate from the best reported results. For MS Marco dataset this refers to MS Marco leaderboard, for the other datasets to the reported benchmarks in the corresponding paper (MacAvaney et al. [13], MacAvaney et al. [14], Yates et al. [28]).

Multiple entity views increase performance to single view: The initial approach, EVA Single, which incorporates entities in documents without a specific focus on the query information, exhibits poor results and even underperforms the baseline TAS BERT. This is due to the inclusion of entirely irrelevant entities within documents during the retrieval process, leading to biased results (see section 3.2.3). However, when considering only entities relevant to the respective query, as in the cases of EVA Single-QA and EVA Multi, a significant increase in effectiveness is observed compared to all baselines, as explained earlier.

KNRM signal provides slight improvement of effectiveness: The comparison of the results for the EVA models with and without the additional KNRM signal shows slightly better performance for the models with the KNRM signal. This is evident in the exemplary results of Figure 8, where the EVA Multi KNRM model achieves an nDCG@10 value of 0.748, slightly higher than the value of 0.733 for the EVA Multi model. A similar observation is made for the EVA Single-QA model. However, the effect is modest, as the EVA Multi approach outperforms the same approach with the additional KNRM signal in the case of the TREC DL 2020 dataset. In conclusion, the impact of introducing entity embeddings is more significant than that of introducing the KNRM signal.

Removing known query assumptions has minor impact on effectiveness, but increases efficiency drastically: EVA Multi and EVA Single-QA demonstrate the most effective outcomes, as evidenced in Figure 8. The crucial distinction between these models lies in the assumption of knowing queries at runtime for EVA Single-QA and the need for pretrained language model inference at runtime (see section 3.2.3). This leads to substantial differences in latency compared to the EVA Multi model. As shown in Table 3, the query latency for the EVA Multi models is 74 ms with KNRM signal and 76 ms without, whereas the latency for both EVA Single-QA models exceeds two seconds. In real-world scenarios, such prolonged latency times are impractical, as users typically expect faster results. However, given that the effectiveness results of EVA Multi and EVA Single-QA only differ marginally, the EVA Multi approach provides a good compromise between efficiency and effectiveness.

5 Discussion & Criticism

Tran and Yates propose an approach that demonstrates the significant improvement of classical dense retrieval methods through the incorporation of entity information. The results, surpassing the baselines TAS BERT and ERNIE, support the effectiveness of their method. Several positive aspects and some limitations of their work can be identified.

5.1 Positives

The approach by Tran and Yates demonstrates superior performance over the baseline methods, as elucidated in the outcomes section. This substantiates the rationale behind the introduction of their approach.

The concepts and elucidations provided by Tran and Yates are marked by simplicity and coherence. The introduced methods (see section 3) have a clear structure. For instance, the choice of aggregation method for embeddings of word tokens and entities, i.e. concatenation of embeddings (see Figure 1), is easier to understand than alternative approaches like max pooling or sum pooling. Based on that, the system could be extended with additional embeddings without much effort.

While many research efforts focus solely on achieving top results on leader boards, Tran and Yates' approach appears to be more oriented towards practical use and sustainability. The EVA Multi-approach offers both high effectiveness and efficiency.

5.2 Negatives

Tran and Yates concentrate solely on the impact of entities in their work, but their analyses reveal that for many queries, entities play no significant role. As shown in Table 2, 43.5 % of all queries in the training data do not contain entities, causing the EVA models being ineffective for such queries.

A further drawback of the approach is its lack of originality. Tran and Yates' propositions are rooted in pre-existing frameworks, with their primary contribution stemming from the combination of ideas from prior authors. As a result, the approach, while intuitive, cannot be classified as groundbreaking. This is evident from the limited citations of their work so far, with only one citation in Kamphuis et al. [10].

5.3 Possible Extensions

The granular structure of Tran and Yates' models allows for possible extensions through the exchange or addition of individual components. For instance, the pre-trained language model used to calculate word-level embeddings could be extended beyond the two baseline models TAS BERT and ERNIE to other, more sophisticated models. In domain-specific use cases, a tailored choice of the pre-trained model might further enhance effectiveness. For example, in a biomedical context, BioBERT (Lee et. al [11]) could be considered.

The embedding component for entities could also be replaced or supplemented with other choices. For example, keyword embeddings, as presented by Gab'ın et al. [6], or structural information, such as that presented by Raman et al. [18], could serve as alternative sources for external embeddings. Furthermore, in the context of HTML files, Guo et al.'s approach [7] could be explored for generating embeddings.

These extensions could address some limitations of the current approach and potentially lead to further improvements in effectiveness and applicability in different domains.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Krisztian Balog. *Entity-oriented search.* Springer Nature, 2018.

[3] Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego, and Salvatore Trani. Dexter: an open source framework for entity linking. In *Proceedings of the sixth international workshop on Exploiting semantic annotations in information retrieval*, pages 17–20, 2013.

[4] Jeffrey Dalton, Laura Dietz, and James Allan. Entity query feature expansion using knowledge base links. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 365–374, 2014.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Jorge Gabín, M Eduardo Ares, and Javier Parapar. Keyword Embeddings for Query Suggestion. In *European Conference on Information Retrieval*, pages 346–360. Springer, 2023.

[7] Yu Guo, Zhengyi Ma, Jiaxin Mao, Hongjin Qian, Xinyu Zhang, Hao Jiang, Zhao Cao, and Zhicheng Dou. Webformer: Pre-training with web pages for information retrieval. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1502–1512, 2022.

[8] Benjamin Heinzerling and Kentaro Inui. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. *arXiv preprint arXiv:2008.09036*, 2020.

[9] Sebastian Hofstätter, Sheng-Chieh Lin, Jheng-Hong Yang, Jimmy Lin, and Allan 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, pages 113–122, 2021.

[10] Chris Kamphuis, Aileen Lin, Siwen Yang, Jimmy Lin, Arjen P de Vries, and Faegheh Hasibi. MMEAD: MS MARCO Entity Annotations and Disambiguations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2817–2825, 2023.

[11] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[12] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908, 2020.

[13] Sean MacAvaney, Andrew Yates, Sergey Feldman, Wei Guo, Yixing Hua, Tom Kenter, Federico Nanni, Bhaskar Mitra, Navid Rekabsaz, and Hamed Zamani. Overview of the TREC 2019 Deep Learning Track. In *Proceedings of The 28th Text REtrieval Conference (TREC 2019)*, 2019.

[14] Sean MacAvaney, Andrew Yates, Sergey Feldman, Wei Guo, Yixing Hua, Tom Kenter, Bhaskar Mitra, Federico Nanni, Navid Rekabsaz, and Hamed Zamani. Overview of the TREC 2020 Deep Learning Track. In *Proceedings of The 29th Text REtrieval Conference (TREC 2020)*, 2020.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26:3111–3119, 2013.

[16] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. *choice*, 2640:660, 2016.

[17] Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. Knowledge enhanced

contextual word representations. *arXiv preprint arXiv:1909.04164*, 2019.

[18] Natraj Raman, Sameena Shah, and Manuela Veloso. Structure and Semantics Preserving Document Representations. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 780–790, 2022.

[19] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, March 2009. ISSN 1554-0669.

[20] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[21] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

[22] Hai Dang Tran and Andrew Yates. Dense Retrieval with Entity Views. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1955–1964, 2022.

[23] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. Word-entity duet representations for document ranking. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, pages 763–772, 2017.

[24] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*, pages 55–64, 2017.

[25] Chenyan Xiong, Russell Power, and Jamie Callan. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*, pages 1271–1279, 2017.

[26] Yang Xu, Gareth JF Jones, and Bin Wang. Query dependent pseudo-

relevance feedback based on wikipedia. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 59–66, 2009.

[27] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia. *arXiv preprint arXiv:1812.06280*, 2018.

[28] Andrew Yates, Sean MacAvaney, Bhaskar Mitra, Navid Rekabsaz, Hamed Zamani, Chenyan Li, Xiang Xu, Zhuyun Dai, Saptarshi Pal, Hui Fang, et al. Overview of the TREC 2020 Deep Learning for Hard Information Retrieval (DLHARD) Track. In *Proceedings of The 29th Text REtrieval Conference (TREC 2020)*, 2020.