Title: Text Representations for Ranking - BOW Encodings

Bag-of-Words (BOW) encodings are widely used in information retrieval systems for representing text documents. In this section, we discuss the use of BOW encodings for ranking documents in neural information retrieval systems.

BOW encodings represent documents as a collection of words, disregarding the order and structure of the text. Each document is represented as a vector, where each dimension corresponds to a unique word in the vocabulary, and the value in each dimension represents the frequency or presence of the word in the document. BOW encodings have been extensively used in traditional information retrieval systems due to their simplicity and effectiveness [REF0].

One of the advantages of BOW encodings is their ability to capture the statistical properties of the text. By considering the frequency or presence of words, BOW encodings can capture the importance of certain terms in a document. This allows for efficient retrieval based on term matching [REF3].

However, BOW encodings have limitations. They do not consider the semantic meaning or context of words, which can lead to ambiguity and lack of precision in retrieval. Additionally, BOW encodings ignore the positional information of words within the document, which can be important for certain retrieval tasks [REF5].

Despite these limitations, BOW encodings have been successfully used in various ranking models. For example, the BM25 model, which is a popular ranking function in information retrieval, utilizes BOW encodings to calculate term weights and document scores [REF9]. BM25 considers factors such as term frequency, document length, and term importance to rank documents [REF8].

In recent years, neural approaches have been proposed to enhance BOW encodings for ranking. These approaches aim to capture the semantic meaning and contextual information of words by incorporating neural network models. For instance, word embeddings, which are dense vector representations of words, can be used to enhance BOW encodings by capturing semantic relationships between words [REF2].

Furthermore, deep learning models, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have been employed to learn more expressive representations from BOW encodings. These models can capture higher-level features and dependencies in the text, leading to improved ranking performance [REF6].

In conclusion, BOW encodings provide a simple and effective representation for ranking documents in neural information retrieval systems. While they have limitations in capturing semantic meaning and positional information, they can be enhanced by incorporating neural network models. The combination of BOW encodings with neural approaches has shown promising results in improving the accuracy and effectiveness of document ranking [REF7].

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: This can much improved by adding the scaling parameter λ, and it can be further improved (only slightly) by changing the log into a sigmoid and tuning the two extra parameters λ and λ . In our work we developed more sophisticated ranking functions integrating several forms of non-textual information and using over a dozen parameters [12, 13]. The optimisation of these parameters is discussed in Section 5. 3.8 Positional Information For the most part the PRF ignores positional information: it cares only about the number of occurrences of a term, but not about their position. There are two important reasons that have held back the PRF from considering positional information: (i) it is extremely hard to develop a sound formal model of relevance which takes into account positional information without exploding the number of parameters, and (ii) position information has been shown to have surprisingly little effect on retrieval accuracy on average. In this section we only give an overview of the existing approaches and discuss the main difficulties.

[REF1] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: This might be a reasonable assumption for traditional ad hoc retrieval, but does not work for all retrieval situations. In some, for example in adaptive filtering , we find it desirable or necessary to arrive at an explicit estimate of the probability of relevance of each considered document. Unfortunately, while346 Development of the Basic Model the above development allows us to serve the ranking purpose well, it is not easily reversible to give us such an explicit estimate. In particular, some of the transformations involved dropping components which would not affect the ranking, but would be required for a good probability estimate. Often, as in the case of the component that we drop at Step (2.3), it would be very difficult to estimate. Thus the above model has to be considerably modified if it is to be used in a situation which requires an explicit probability. This issue is not discussed further in the present survey.3 Derived Models

[REF2] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: When it is well understood, the PRF model can provide a solid ground on which to analyse new IR problems and derive new solutions. 384References S. Agarwal, C. Cortes, and R. Herbrich, eds., Proceedings of the NIPS 2005 Workshop on Learning to Rank, 2005. G. Amati, C. J. van Rijsbergen, and C. Joost, “Probabilistic models of information retrieval based on measuring the divergence from randomness,” ACM Transactions on Information Systems, vol. 20, no. 4, pp. 357–389, 2002. M. M. Beaulieu, M. Gatford, X. Huang, S. E. Robertson, S. Walker, and P. Williams, “Okapi at TREC-5,” The Fifth Text Retrieval Conference (TREC5).

[REF3] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF11\_73a76dd71abfbd29dbba4ea034ab52284626aa71.pdf Title: A Language Modeling Approach to Information Retrieval Chunk of text: For this reason, we do not use the standard tf and idf scores. In addition, length normalization is implicit in the calculation of the probabilities and does not have to be done in an ad hoc manner. The remainder of the paper is organized as follows. In section 2 we review some existing retrieval models. Section 3 describes a language modeling approach that closely parallels the standard approach to IR. Section 4 shows the effectiveness of this model empirically. Finally we offer concluding remarks and describe future directions of this work.

[REF4] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: i + fi)��3.8 Positional Information 367 • Rational: fi/(λ i + fi) • Sigmoid: [λ i + exp(−fi λ i )]−1 This development can explain for example why simple scoring functions such as BM25F(q,d)+log(P ageRank(d)) may work well in practice for Web Search retrieval. This can much improved by adding the scaling parameter λ, and it can be further improved (only slightly) by changing the log into a sigmoid and tuning the two extra parameters λ and λ .

[REF5] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: We end this section with a brief discussion of why position information may not be as important as it may seem at first view. It is sobering to see how hard it has been in the past to effectively use proximity in IR experiments. All the works referenced in this section claim statistically significant improvements over non-positional baselines, but the improvements reported are small. We believe this is specially the case for small collections and high recall situations (typical of academic IR evaluations), since position information is a precision enhancement technique. But even in large collections with high-precision requirements (such as realistic Web Search evaluations) the gains observed are small. Why is this? We do not know of theoretical or empirical studies about this, but we propose here two hypotheses.

[REF6] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: Any discrete property with a natural zero can be dealt with using the Wi form of the weight — if we want to include a property without such a natural zero, we need to revert to the Ui form. We note also that both forms are simple linear models — the combination of evidence from the different query terms is just by summation. This is not in itself an assumption — it arises naturally from the more basic assumptions of the model. In the sections which follow, we define various instantiations of this basic sum-of-weights scoring model. 2.5 A Note on Probabilities and Rank Equivalence One consequence of our reliance on the probability ranking principle is that we are enabled to make the very cavalier transformations discussed above, on the basis that the only property we wish to preserve is the rank order of documents. This might be a reasonable assumption for traditional ad hoc retrieval, but does not work for all retrieval situations. In some, for example in adaptive filtering , we find it desirable or necessary to arrive at an explicit estimate of the probability of relevance of each considered document.

[REF7] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: The model estimates the probability of relevance of document and query jointly: P(Q,D |Rel). This is done by a Markov Random Field (MRF) which can take into account term-positions in a natural way. The MRF can use any appropriately defined potentials: while the original work used LM-derived potentials, BM25-like potentials were used in . However, even when using BM25-like potentials, we cannot call this model an extension of3.9 Open Source Implementations of BM25 and BM25F 369 the PRF, since it models a different distribution: P(D,Q|Rel) instead of the posterior P(Rel|D,Q). We end this section with a brief discussion of why position information may not be as important as it may seem at first view. It is sobering to see how hard it has been in the past to effectively use proximity in IR experiments.

[REF8] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: Again, this has led to one of the most successful Web-search and corporate-search algorithms, BM25F. This work presents the PRF from a conceptual point of view, describing the probabilistic modelling assumptions behind the framework and the different ranking algorithms that result from its application: the binary independence model, relevance feedback models, BM25 and BM25F. It also discusses the relation between the PRF and other statistical models for IR, and covers some related topics, such as the use of non-textual features, and parameter optimisation for models with free parameters.1 Introduction This monograph addresses the classical probabilistic model of information retrieval. The model is characterised by including a specific notion of relevance, an explicit variable associated with a query–document pair, normally hidden in the sense of not observable. The model revolves around the notion of estimating a probability of relevance for each pair, and ranking documents in relation to a given query in descending order of probability of relevance. The best-known instantiation of the model is the BM25 term-weighting and document-scoring function. The model has been developed in stages over a period of about 30 years, with a precursor in 1960. A few of the main references are as follows: [30, 44, 46, 50, 52, 53, 58]; other surveys of a range of probabilistic approaches include [14, 17].

[REF9] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-BOW\_Encodings/BIBREF10\_47ced790a563344efae66588b5fb7fe6cca29ed3.pdf Title: The Probabilistic Relevance Framework: BM25 and Beyond Chunk of text: b dl avdl + tf wRSJ i (3.15) This is the classic BM25 term-weighting and document-scoring function. As with all term-document weights defined in this survey, the full document score is obtained by summing these term-weights over the (original or expanded) set of query terms. 3.5 Uses of BM25 In order to use BM25 as a ranking function for retrieval, we need to choose values for the internal parameters b and k1, and also instantiate RSJ.

........................................................................................................................................................................................................

Title: Text Representations for Ranking - LTR Features

In the field of Neural Information Retrieval, learning to rank (LTR) has gained significant attention as a powerful approach for improving the effectiveness of ranking models in information retrieval systems [REF0]. LTR involves using machine learning techniques to learn an appropriate combination of features that can effectively rank documents for a given query [REF0]. Major search engines have been reported to deploy ranking models consisting of hundreds of features [REF0]. However, the deployment of LTR in real settings has not been extensively discussed in the literature [REF0].

One important aspect of LTR is the representation of text documents for ranking purposes. The choice of document representation can have a significant impact on the effectiveness of the learned ranking model. In the literature, there is a lack of clarity regarding the properties of an effective sample of top-ranked documents for a given query [REF0]. For example, it is not clear what document representation should be used when generating the sample, such as whether anchor text should be included or not [REF0]. Additionally, the number of documents in the sample is also an important consideration [REF0].

Previous studies have investigated the impact of different factors on the effectiveness of LTR models. For instance, experiments on the NAV06 navigational query set have explored the effects of sample size, document representation, learning evaluation measures, and rank cutoffs [REF1]. The results showed that the choice of sample document representation, such as using anchor text in addition to the document content, can significantly impact the retrieval effectiveness, especially for smaller sample sizes [REF1]. It was found that a sample size of at least 1500 documents guarantees effective retrieval for all query sets using this representation [REF3].

Furthermore, the choice of learning evaluation measures within the loss function of listwise learning to rank techniques has been shown to affect the effectiveness of the learned models [REF3]. Measures such as Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP) were found to be more effective than Expected Reciprocal Rank (ERR) [REF3]. The rank cutoff of the learning evaluation measure, particularly for precision, was also found to impact the effectiveness of the learned models [REF4].

In summary, the choice of text representations for ranking plays a crucial role in the effectiveness of LTR models. The inclusion of anchor text in the document representation has been shown to improve retrieval effectiveness, especially for smaller sample sizes [REF1][REF3]. The number of documents in the sample is also an important consideration, with larger samples generally leading to better performance [REF8]. Additionally, the choice of learning evaluation measures and their rank cutoff

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: Introduction Learning to rank (Liu 2009) is gaining increasing attention in information retrieval (IR), with machine learning techniques being used to learn an appropriate combination of features into an effective ranking model. An increasing amount of research is devoted to developing efficient and effective learning techniques, while major search engines reportedly deploy ranking models consisting of hundreds of features (Pederson 2010; Segalovich 2010). Nevertheless, the manner in which learning to rank is deployed within real settings has not seen published discussion. For instance, learning to rank involves the use of a sample of top-ranked documents for a given query (Liu 2009), which are then re-ranked by the learned model before display to the user. However, in the literature, the properties of an effective sample are not clear. Indeed, despite his thorough treatment of existing learning to rank techniques, Liu (2009) does not address in detail how the sample should be made within an existing deployment, what document representation should be deployed when generating the sample (e.g. in addition to the body of the document, should anchor text be included or not?), nor how many documents it should contain. Typically, a standard weighting model, such as BM25 (Robertson et al 1992), is used to rank enough documents to obtain sufficient recall.

[REF1] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: For the NAV06 navigational query set, we report mean reciprocal rank (MRR). Aside from the deployed learning to rank technique, there are four factors in our experiments that we defined in Section 3, namely: the size of the sample; the document representation used to generate the sample; the learning evaluation measure using within the loss function of the AFS and AdaRank listwise learning to rank techniques; and the rank cutoff of the learning evaluation measure. Indeed, as per the methodology prescribed in Section 3, we differentiate between the test evaluation measure, which remains fixed, and the learning evaluation measure used by a listwise learning to rank technique, which we vary. The settings for each of the factors in our experiments are as follows: – Sample Document Representation - For GOV, BM25 with anchor text, as provided by LETOR v3.0. For CW09B, DPH with or without anchor text, as illustrated in Figure 4. – Sample Size - We experiment with different sample sizes, up to the maximum permitted by the original samples of size 1000 for GOV and 5000 for CW09B: GOV = {10, 20, 50, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000};24 Craig Macdonald et al. Table 6 All factors in our experiments. GOV CW09B Learners GBRT, RankBoost, RankNet, LambdaMART, AFS, AdaRank Sample Document Model BM25 DPH & Representation with anchor text with/without anchor text Sample Size {10, 20, 50, 100, 200, 300, 400, {10, 20, 50, 100, 500, 1000, 500, 600, 700, 800, 900, 1000} 1500, 2000, 3000, 4000, 5000} Learning Evaluation Measure P,MAP,MRR,NDCG,ERR Learning Evaluation Measure {10, 20, 50, 100, 200, 300, 400, {10, 20, 50, 100, 500, 1000 Cutoffs 500, 600, 700, 800, 900, 1000} 1500, 2000, 3000, 4000, 5000} CW09B = {10, 20, 50, 100, 500, 1000, 1500, 2000, 3000, 4000, 5000}.

[REF2] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF16\_63aaf12163fe9735dfe9a69114937c4fa34f303a.pdf Title: Learning to Rank using Gradient Descent Chunk of text: We present results on toy data and on data gathered from a commercial internet search engine. For the latter, the data takes the form of 17,004 queries, and for each query, up to 1000 returned documents, namely the top docu- ∗Current affiliation: Google, Inc.Learning to Rank using Gradient Descent ments returned by another, simple ranker. Thus each query generates up to 1000 feature vectors. Notation: we denote the number of relevance levels (or ranks) by N, the training sample size by m, and the dimension of the data by d. 2. Previous Work RankProp (Caruana et al., 1996) is also a neural net ranking model. RankProp alternates between two phases: an MSE regression on the current target values, and an adjustment of the target values themselves to reflect the current ranking given by the net.

[REF3] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: For the navigational query set on ClueWeb09, we found that it was important to use anchor text in the sampling document representation to ensure effective retrieval at sample sizes smaller than 5000 documents – indeed, a sample size of at least 1500 documents guarantees effective retrieval for all query sets using this representation. These results suggest that deep samples are necessary for effective retrieval in large Web corpora, indeed deeper than some recent learning to rank test collections such as those listed in Table 2. In addition, our experiments also showed that the effectiveness of learned models are generally dependent on the sample size (Hypothesis 1), partially dependent on the type of information need and the sample document representation (Hypothesis 2), and partially dependent on the choice of the learning to rank technique and the sample size (Hypothesis 3). With respect to our second research theme addressing how the loss function for listwise learning to rank techniques should be defined – i.e. the choice of learning evaluation measures deployed by listwise learning to rank techniques – we found that the choice of the learning evaluation measure can indeed have an impact upon the effectiveness of the resulting learned model (Hypothesis 4), particularly for informational needs. Indeed, our results show that NDCG and MAP are the most effective learning evaluation measures, while the less informative ERR was not as effective, even when the test performance is evaluated by ERR. For the learning evaluation measure rank cutoff (Hypothesis 5), we only found the effectiveness of learned models to be markedly impacted by the rank cutoff for the precision measure. Finally, for our third research theme, we showed that while there is no dependence between the learning measure cutoff and the sample size in terms of the effectiveness of the learned model (Hypothesis 6), the weights of the selected features can markedly differ between small and large samples, and between small and large learning evaluation measure cutoffs.

[REF4] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: For the learning evaluation measure rank cutoff (Hypothesis 5), we only found the effectiveness of learned models to be markedly impacted by the rank cutoff for the precision measure. Finally, for our third research theme, we showed that while there is no dependence between the learning measure cutoff and the sample size in terms of the effectiveness of the learned model (Hypothesis 6), the weights of the selected features can markedly differ between small and large samples, and between small and large learning evaluation measure cutoffs. To summarise our empirical findings for applying learning to rank on a large Web corpus such as ClueWeb09, where evaluation is conducted using a measure such as NDCG@20 or MRR, our results suggest that the sample should contain no less than 1500 documents, and be created using a document representation that considers anchor text in addition to the content of the document, so as to ensure effective retrieval for both informational and navigational information needs (Section 5.1.4). If a listwise learning to rank technique is used to obtain the learned model, then our results suggest that NDCG@10 represents a suitably informative learning evaluation measure to achieve an effective learned46 Craig Macdonald et al. model (Section 5.2.3). Lastly, the importance of different classes of features within a learned model are dependent on both the sample size and the rank cutoff of the learning evaluation measure (Section 5.3.3). This work used a total of six learning to rank techniques that are widely implemented or freely available, which are representative of the various families (pointwise, pairwise and listwise), as well as of different types of learned models (linear combination, neural network, or regression tree). The used learning to rank techniques includes the state-of-the-art LambdaMART technique, which won the Yahoo!

[REF5] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: Similarly, we split the 150 NAV06 queries into equal sets, to form a single fold with separate training, validation and testing query subsets. Table 5 also records the test evaluation measure used to test the effectiveness of the learned models on each query set. Indeed, for the GOV query sets, the measure used for each query set matches the official measure used by the corresponding TREC Web track (Craswell and Hawking 2004). For the CW09B query sets, NDCG@20 was used for the evaluation measure in TREC 2009 (Clarke et al 2010), while for TREC 2010, ERR@20 was used (Clarke et al 2011). In the following experiments, we use both measures, and for completeness, we additionally use MAP for both WT09 and WT10 query sets. For the NAV06 navigational query set, we report mean reciprocal rank (MRR). Aside from the deployed learning to rank technique, there are four factors in our experiments that we defined in Section 3, namely: the size of the sample; the document representation used to generate the sample; the learning evaluation measure using within the loss function of the AFS and AdaRank listwise learning to rank techniques; and the rank cutoff of the learning evaluation measure.

[REF6] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: It represents a larger and more thorough study than any currently present in the literature, investigating practical issues that arise when deploying learning to rank. In particular, we define three research themes, with corresponding hypotheses and research questions. In the first theme, we address the size and constitution of the sample for learning to rank (when to stop ranking). Moreover, in the second research theme, we address the choice of the learning evaluation measure and the corresponding rank cutoff for listwise learning to rank techniques (how to evaluate the learned models within the loss function of listwise learning to rank techniques). Lastly, our final research theme investigates the dependence between the sample size and the learning evaluation measure rank cutoff. Overall, we found that when to stop ranking – the smallest effective sample – varied according to several factors: the information need, the evaluation measure used to test the models, and the presence of anchor text in the documentThe whens and hows of learning to rank for web search 45 representation used for sampling. In particular, from Table 10, we found that the smallest effective sample size was 10-50 documents for navigational information needs on the GOV (LETOR) test collection, while 400 documents were necessary for the topic distillation query set.

[REF7] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: In summary, we find that Hypothesis 2 is partially validated: the impact of the sample size can depend on the type of information need, while the presence of anchor text is important for assuring the effectiveness of smaller sample sizes for navigational queries. Hypothesis 3 stipulates that the effectiveness of learned models depends on both the choice of the learning to rank technique and the sample size. To analyse this hypothesis, we use the last column of Table 8, which denotes significant dependencies on both of these independent variables. In general, we observe significant dependencies between effectiveness and the learning to rank technique only for the HP04 and NP04 query sets, as well as WT09a. On inspection of the corresponding figures for these query sets (namely Figures 5(a) & (b) and Figure 7(a), (c) & (e)), this observation can be explained as follows: For these query sets and corresponding test evaluation measures, there are learning to rank techniques that markedly degrade in performance forThe whens and hows of learning to rank for web search 33 Table 9 Recall of samples of size 1000 and 5000 for the ClueWeb09 query sets, as the presence of anchor text in the sample document representation is varied. Query Set ↓ 1000 5000 Anchor Text → ✗ ✔ ✗ ✔

[REF8] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: Overall, we found that when to stop ranking – the smallest effective sample – varied according to several factors: the information need, the evaluation measure used to test the models, and the presence of anchor text in the documentThe whens and hows of learning to rank for web search 45 representation used for sampling. In particular, from Table 10, we found that the smallest effective sample size was 10-50 documents for navigational information needs on the GOV (LETOR) test collection, while 400 documents were necessary for the topic distillation query set. For the TREC Web track query sets of mixed types of information needs on the much larger ClueWeb09 corpus, samples with as little as 20 documents are sufficient for effective ERR@20 performances. Some techniques and query sets were shown to require larger samples (up to 2000 documents) for effective NDCG@20. Furthermore, for an effective MAP performance, samples of 2000 documents were shown to be necessary for all learning to rank techniques. For the navigational query set on ClueWeb09, we found that it was important to use anchor text in the sampling document representation to ensure effective retrieval at sample sizes smaller than 5000 documents – indeed, a sample size of at least 1500 documents guarantees effective retrieval for all query sets using this representation. These results suggest that deep samples are necessary for effective retrieval in large Web corpora, indeed deeper than some recent learning to rank test collections such as those listed in Table 2.

[REF9] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-LTR\_Features/BIBREF12\_008f1d2741ebef51e6400686b050e046455b52fb.pdf Title: The Whens and Hows of Learning to Rank for Web Search Chunk of text: As discussed in Section 2.1.2, we assume that the learned model will be deployed on a single retrieval system, with the sample generated by using a standard weighting model. For learning, documents in the sample are labelled using high quality relevance assessments (e.g. from TREC) that are already available. We investigate the research questions and hypotheses of our three research themes across different scenarios, including different types of information needs, learning to rank techniques, and corpora. To investigate each of our research questions, we perform empirical experiments using multiple query sets on several learning to rank test collections. In particular, query sets are paired into training and testing sets with no overlap of queries. For instance, for a given query set (HP04, NP04 etc.), LETOR v3.0 GOV prescribes 5 folds, each with separate training and testing query sets. On a given testing query set, the test evaluation measure and its rank cutoff are pre-defined - to use the experimental design terminology, this is the dependent variable.

........................................................................................................................................................................................................

Title: Text Representations for Ranking - Word Embeddings

Word embeddings have become a popular approach for representing text in neural information retrieval systems. These embeddings capture the semantic and syntactic relationships between words by mapping them to continuous vector representations in a high-dimensional space. This section explores the use of word embeddings for ranking in neural information retrieval.

One common approach is to learn word embeddings using unsupervised methods. These methods aim to capture the meaning of words based on their co-occurrence patterns in large text corpora. For instance, Mikolov et al. introduced the Skip-gram model, which learns high-quality vector representations of words from unstructured text data [REF9]. The Skip-gram model efficiently captures the semantic relationships between words by predicting the context words given a target word. The resulting word embeddings can be used to measure the similarity between words and to capture their semantic relatedness.

Word embeddings can also be extended to represent larger units of text, such as sentences or paragraphs. Sentence embeddings have been generated by ranking candidate next sentences, generating next sentence words given a representation of the previous sentence, or using denoising autoencoder derived objectives [REF0]. These sentence embeddings can capture the contextual information and semantic meaning of sentences, enabling more accurate ranking of documents based on their relevance to a query.

Another approach to word embeddings is the use of contextual word embeddings. Contextual word embeddings, such as ELMo (Embeddings from Language Models), extract context-sensitive features from both left-to-right and right-to-left language models [REF0]. The contextual representation of each token is obtained by concatenating the left-to-right and right-to-left representations. This approach has shown significant improvements in various natural language processing tasks, including question answering, sentiment analysis, and named entity recognition [REF0].

Evaluation of word embeddings is crucial to assess their quality and effectiveness. Various evaluation schemes have been proposed, including distance or angle-based evaluations, as well as word analogy evaluations [REF1]. The word analogy evaluation measures the ability of word embeddings to capture the finer structure of the word vector space by examining the dimensions of difference between word vectors. This evaluation scheme favors models that produce dimensions of meaning, thereby capturing the multi-clustering idea of distributed representations [REF1].

In addition to the intrinsic evaluation of word embeddings, they have been successfully applied to various information retrieval tasks. For instance, word embeddings have been used as features in document classification, information retrieval, question answering, named entity recognition, and parsing [REF1]. These applications leverage the semantic relationships captured by word embeddings to improve the performance of the respective tasks.

In summary, word embeddings provide a powerful representation for ranking in neural information retrieval systems. They capture the semantic and syntactic relationships between words, sentences, and paragraphs, enabling more accurate ranking of documents based on their relevance to a query. The evaluation of word embeddings is crucial to assess their quality, and they have been successfully applied to various information retrieval tasks.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF21\_df2b0e26d0599ce3e70df8a9da02e51594e0e992.pdf Title: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding Chunk of text: These approaches have been generalized to coarser granularities, such as sentence embeddings (Kiros et al., 2015; Logeswaran and Lee, 2018) or paragraph embeddings (Le and Mikolov, 2014). To train sentence representations, prior work has used objectives to rank candidate next sentences (Jernite et al., 2017; Logeswaran and Lee, 2018), left-to-right generation of next sentence words given a representation of the previous sentence (Kiros et al., 2015), or denoising autoencoder derived objectives (Hill et al., 2016). ELMo and its predecessor (Peters et al., 2017, 2018a) generalize traditional word embedding research along a different dimension. They extract context-sensitive features from a left-to-right and a right-to-left language model. The contextual representation of each token is the concatenation of the left-to-right and right-to-left representations. When integrating contextual word embeddings with existing task-specific architectures, ELMo advances the state of the art for several major NLP benchmarks (Peters et al., 2018a) including question answering (Rajpurkar et al., 2016), sentiment analysis (Socher et al., 2013), and named entity recognition (Tjong Kim Sang and De Meulder, 2003). Melamud et al.

[REF1] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF20\_f37e1b62a767a307c046404ca96bc140b3e68cb5.pdf Title: GloVe: Global Vectors for Word Representation Chunk of text: Introduction Semantic vector space models of language represent each word with a real-valued vector. These vectors can be used as features in a variety of applications, such as information retrieval (Manning et al., 2008), document classification (Sebastiani, 2002), question answering (Tellex et al., 2003), named entity recognition (Turian et al., 2010), and parsing (Socher et al., 2013). Most word vector methods rely on the distance or angle between pairs of word vectors as the primary method for evaluating the intrinsic quality of such a set of word representations. Recently, Mikolov et al. (2013c) introduced a new evaluation scheme based on word analogies that probes the finer structure of the word vector space by examining not the scalar distance between word vectors, but rather their various dimensions of difference. For example, the analogy “king is to queen as man is to woman” should be encoded in the vector space by the vector equation king − queen = man − woman. This evaluation scheme favors models that produce dimensions of meaning, thereby capturing the multi-clustering idea of distributed representations (Bengio, 2009). The two main model families for learning word vectors are: 1) global matrix factorization methods, such as latent semantic analysis (LSA) (Deerwester et al., 1990) and 2) local context window methods, such as the skip-gram model of Mikolov et al. (2013c).

[REF2] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF20\_f37e1b62a767a307c046404ca96bc140b3e68cb5.pdf Title: GloVe: Global Vectors for Word Representation Chunk of text: 6B 57.2 65.6 68.2 57.0 32.5 SG† 6B 62.8 65.2 69.7 58.1 37.2 GloVe 6B 65.8 72.7 77.8 53.9 38.1 SVD-L 42B 74.0 76.4 74.1 58.3 39.9 GloVe 42B 75.9 83.6 82.9 59.6 47.8 CBOW∗ 100B 68.4 79.6 75.4 59.4 45.5 L model on this larger corpus. The fact that this basic SVD model does not scale well to large corpora lends further evidence to the necessity of the type of weighting scheme proposed in our model. Table 3 shows results on five different word similarity datasets. A similarity score is obtained from the word vectors by first normalizing each feature across the vocabulary and then calculating the cosine similarity. We compute Spearman’s rank correlation coefficient between this score and the human judgments. CBOW∗ denotes the vectors available on the word2vec website that are trained with word and phrase vectors on 100B words of news data.

[REF3] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF23\_cd18800a0fe0b668a1cc19f2ec95b5003d0a5035.pdf Title: Improving Language Understanding by Generative Pre-Training Chunk of text: . We used learned position embeddings instead of the sinusoidal version proposed in the original work. We use the ftfy library2 to clean the raw text in BooksCorpus, standardize some punctuation and whitespace, and use the spaCy tokenizer.3 Fine-tuning details Unless specified, we reuse the hyperparameter settings from unsupervised pre-training. We add dropout to the classifier with a rate of 0.1. For most tasks, we use a learning rate of 6.25e-5 and a batchsize of 32. Our model finetunes quickly and 3 epochs of training was sufficient for most cases. We use a linear learning rate decay schedule with warmup over 0.2% of training.

[REF4] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF20\_f37e1b62a767a307c046404ca96bc140b3e68cb5.pdf Title: GloVe: Global Vectors for Word Representation Chunk of text: Our results using the word2vec tool are somewhat better than most of the previously published results. This is due to a number of factors, including our choice to use negative sampling (which typically works better than the hierarchical softmax), the number of negative samples, and the choice of the corpus. We demonstrate that the model can easily be trained on a large 42 billion token corpus, with a substantial corresponding performance boost. We note that increasing the corpus size does not guarantee improved results for other models, as can be seen by the decreased performance of the SVD7We also investigated several other weighting schemes for transforming X; what we report here performed best. Many weighting schemes like PPMI destroy the sparsity of X and therefore cannot feasibly be used with large vocabularies. With smaller vocabularies, these information-theoretic transformations do indeed work well on word similarity measures, but they perform very poorly on the word analogy task. Table 3: Spearman rank correlation on word similarity tasks.

[REF5] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF19\_892e53fe5cd39f037cb2a961499f42f3002595dd.pdf Title: Bag of Tricks for Efficient Text Classification Chunk of text: We tune the hyperparameters on the validation set and observe that using n-grams up to 5 leads to the best performance. Unlike Tang et al. (2015), fastText does not use pre-trained word embeddings, which can be explained the 1% difference in accuracy. Model Yelp’13 Yelp’14 Yelp’15 IMDB SVM+TF 59.8 61.8 62.4 40.5 CNN 59.7 61.0 61.5 37.5 Conv-GRNN 63.7 65.5 66.0 42.5 LSTM-GRNN 65.1 67.1 67.6 45.3 fastText 64.2 66.2 66.6 45.2 Table 3: Comparision with Tang et al. (2015). The hyperparameters are chosen on the validation set. We report the test accuracy.

[REF6] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF20\_f37e1b62a767a307c046404ca96bc140b3e68cb5.pdf Title: GloVe: Global Vectors for Word Representation Chunk of text: We compute Spearman’s rank correlation coefficient between this score and the human judgments. CBOW∗ denotes the vectors available on the word2vec website that are trained with word and phrase vectors on 100B words of news data. GloVe outperforms it while using a corpus less than half the size. Table 4 shows results on the NER task with the CRF-based model. The L-BFGS training terminates when no improvement has been achieved on the dev set for 25 iterations. Otherwise all configurations are identical to those used by Wang and Manning (2013). The model labeled Discrete is the baseline using a comprehensive set of discrete features that comes with the standard distribution of the Stanford NER model, but with no word vector features.

[REF7] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF18\_87f40e6f3022adbc1f1905e3e506abad05a9964f.pdf Title: Distributed Representations of Words and Phrases and their Compositionality Chunk of text: The product works here as the AND function: words that are assigned high probabilities by both word vectors will have high probability, and the other words will have low probability. Thus, if “Volga River” appears frequently in the same sentence together with the words “Russian” and “river”, the sum of these two word vectors will result in such a feature vector that is close to the vector of “Volga River”. 6 Comparison to Published Word Representations Many authors who previously worked on the neural network based representations of words have published their resulting models for further use and comparison: amongst the most well known authors are Collobert and Weston , Turian et al. , and Mnih and Hinton . We downloaded their word vectors from the web3 .

[REF8] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF18\_87f40e6f3022adbc1f1905e3e506abad05a9964f.pdf Title: Distributed Representations of Words and Phrases and their Compositionality Chunk of text: The subsampling of the frequent words improves the training speed several times and makes the word representations significantly more accurate. It can be argued that the linearity of the skip-gram model makes its vectors more suitable for such linear analogical reasoning, but the results of Mikolov et al. also show that the vectors learned by the standard sigmoidal recurrent neural networks (which are highly non-linear) improve on this task significantly as the amount of the training data increases, suggesting that non-linear models also have a preference for a linear structure of the word representations. 4 Learning Phrases As discussed earlier, many phrases have a meaning that is not a simple composition of the meanings of its individual words. To learn vector representation for phrases, we first find words that appear frequently together, and infrequently in other contexts. For example, “New York Times” and “Toronto Maple Leafs” are replaced by unique tokens in the training data, while a bigram “this is” will remain unchanged. 1code.google.com/p/word2vec/source/browse/trunk/questions-words.txt 5Newspapers New York New York Times Baltimore Baltimore Sun San Jose San Jose Mercury News Cincinnati Cincinnati Enquirer NHL Teams Boston Boston Bruins Montreal Montreal Canadiens Phoenix Phoenix Coyotes Nashville Nashville Predators NBA Teams Detroit Detroit Pistons Toronto Toronto Raptors Oakland Golden State Warriors

[REF9] - paperID: ./papers\_pdf/paper\_section/Text\_Representations\_for\_Ranking-Word\_Embeddings/BIBREF18\_87f40e6f3022adbc1f1905e3e506abad05a9964f.pdf Title: Distributed Representations of Words and Phrases and their Compositionality Chunk of text: Introduction Distributed representations of words in a vector space help learning algorithms to achieve better performance in natural language processing tasks by grouping similar words. One of the earliest use of word representations dates back to 1986 due to Rumelhart, Hinton, and Williams . This idea has since been applied to statistical language modeling with considerable success . The follow up work includes applications to automatic speech recognition and machine translation [14, 7], and a wide range of NLP tasks [2, 20, 15, 3, 18, 19, 9]. Recently, Mikolov et al. introduced the Skip-gram model, an efficient method for learning highquality vector representations of words from large amounts of unstructured text data.

........................................................................................................................................................................................................

Title: Interaction-focused Systems - Convolutional Neural Networks

Convolutional Neural Networks (CNNs) have gained significant attention in the field of neural information retrieval, particularly in interaction-focused systems. These systems aim to capture the interactions between queries and documents to improve relevance ranking. The success of CNNs in this context can be attributed to their ability to learn hierarchical interaction patterns and match query-document pairs effectively [REF1].

One of the key advantages of interaction-focused models is their ability to learn query-document matching patterns from word-level interactions. For instance, models like ARC-II and MatchPyramid utilize hierarchical CNNs to build interactions between word embeddings of two texts [REF1]. These models have shown effectiveness in matching tweet-retweet and question-answer pairs [REF1]. By leveraging the power of deep neural networks, these models can capture complex interaction patterns and improve the retrieval performance.

In interaction-focused models, the architecture typically consists of a simple mapping function, denoted as Φ, which builds local interactions between texts based on basic representations [REF3]. This is followed by a deep neural network, denoted as F, which learns the hierarchical interaction patterns for matching [REF3]. For example, DeepMatch uses a feed-forward neural network powered by a topic model over the word interaction matrix [REF3]. ARC-II and MatchPyramid, on the other hand, employ CNNs over the interaction matrix between word vectors from the two texts [REF3]. This hierarchical deep architecture allows the models to effectively capture and exploit the local interaction information [REF3].

To further enhance the performance of interaction-focused models, various techniques have been proposed. For instance, the Term Gating Network is employed to explicitly model the importance of query terms [REF4]. This network assigns an aggregation weight to each query term, controlling its contribution to the final relevance score [REF4]. By using the softmax function as the gating function, the model can effectively weigh the relevance scores of different query terms [REF4].

While interaction-focused models have shown promise, there have been challenges in applying deep learning to ad-hoc retrieval tasks. Deep models have been successful in other domains, such as computer vision and natural language processing, but their impact on information retrieval has been limited [REF7]. Traditional ranking models, such as language models and BM25, have often outperformed deep models in ad-hoc retrieval tasks [REF9]. However, with the advancements in interaction-focused systems and the utilization of CNNs, there is potential for deep learning to make significant contributions to neural information retrieval.

In summary, interaction-focused systems that leverage Convolutional Neural Networks have shown promise in capturing and exploiting the interactions between queries and documents. These models learn hierarchical interaction patterns and effectively match query-document pairs. Techniques like the Term Gating Network further enhance the performance by explicitly modeling the importance of query terms. While challenges remain in applying deep learning to ad-hoc retrieval tasks, the advancements in interaction-focused systems provide opportunities for deep models to improve the effectiveness of neural information retrieval.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF27\_fc3384d631f5e2b2a9d66623d4d3e1d28b96dee7.pdf Title: Convolutional Neural Networks for Soft-Machting N-Grams in Ad-hoc Search Chunk of text: INTRODUCTION A recent success of neural methods in information retrieval (neural IR) is the development of interaction based models [13, 21, 29]. Interaction based models thrive with encoding word-word translations using word embeddings, and utilizing new pooling methods to beer summarize the word translations into ranking signals [11, 13, 29]. Learned end-to-end from user feedbacks [23, 29], the word embeddings can encode so matches tailored for relevance ranking, which has signicant advantages over traditional feature-based methods [29, 30]. ese initial successes of neural IR were mainly from so matching individual words. On the other Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for prot or commercial advantage and that copies bear this notice and the full citation on the rst page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permied.

[REF1] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF26\_ea738439b880ad033ff01602ea52d04b366d0d37.pdf Title: End-to-End Neural Ad-hod Ranking with Kernel Pooling Chunk of text: For example, DSSM and its convolutional version CDSSM map words to leer-tri-grams, embed query and documents using neural networks built upon the leer-tri-grams, and rank documents using their embedding similarity with the query. e interaction based neural models, on the other hand, learn query-document matching paerns from word-level interactions. For example, ARC-II and MatchPyramid build hierarchical Convolutional Neural Networks (CNN) on the interactions of two texts’ word embeddings; they are eective in matching tweetretweet and question-answers . e Deep Relevance Matching Model (DRMM) uses pyramid pooling (histogram)

[REF2] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF28\_32e7f0863e7c56cfced89abedaee46e2288bc127.pdf Title: PACRR: A Position-Aware Neural IR Model for Relevance Matching Chunk of text: Introduction Despite the widespread use of deep neural models across a range of linguistic tasks, to what extent such models can improve information retrieval (IR) and which components a deep neural model for IR should include remain open questions. In ad-hoc IR, the goal is to produce a ranking of relevant documents given an open-domain (“ad hoc”) query and a document collection. A ranking model thus aims at evaluating the interactions between different documents and a query, assigning higher scores to documents that better match the query. Learning to rank models, like the recent IRGAN model (Wang et al., 2017), rely on handcrafted features to encode query document interactions, e.g., the relevance scores from unsupervised ranking models. Neural IR models differ in that they extract interactions directly based on the queries and documents. Many early neural IR models can be categorized as semantic matching models, as they embed both queries and documents into a low-dimensional space, and then assess their similarity based on such dense representations. Examples in this regard include DSSM (Huang et al., 2013) and DESM (Mitra et al., 2016).

[REF3] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF25\_d51ed05fd05b9d222427a05a87ed88217447b44f.pdf Title: A Deep Relevance Matching Model for Ad-hoc Retrieval Chunk of text: Without loss of generality, all the model architectures of representationfocused models can be viewed as a Siamese (symmetric) architecture over the text inputs, as shown in Figure 1(a). The second one, the interaction-focused model, first builds the local interactions between two texts based on some basic representations, and then uses deep neural networks to learn the hierarchical interaction patterns for matching. In this approach, Φ is usually a simple mapping function while F is a complex deep model. For example, in DeepMatch , Φ simply maps each text to a sequence of words, while F is a feed forward neural network powered by a topic model over the word interaction matrix. In ARC-II and MatchPyramid , Φ maps each text to a sequence of word vectors,while F is a CNN over the interaction matrix between word vectors from the two texts. Without loss of generality, all the model architectures of interaction-focused models can be viewed as a hierarchical deep architecture over the local interaction matrix, as shown in Figure 1(b).

[REF4] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF25\_d51ed05fd05b9d222427a05a87ed88217447b44f.pdf Title: A Deep Relevance Matching Model for Ad-hoc Retrieval Chunk of text: Term Gating Network: One significant difference of our model from existing interaction-focused models is that we employ a joint deep architecture at the query term level. In this way, our model can explicitly model query term importance. This is achieved by using the term gating network, which produces an aggregation weight for each query term controlling how much the relevance score on that query term contributes to the final relevance score. Specifically, we employ the softmax function as the gating function. gi = exp(wgx (q) i ) PM j=1 exp(wgx (q) j ) , i = 1, . .

[REF5] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF26\_ea738439b880ad033ff01602ea52d04b366d0d37.pdf Title: End-to-End Neural Ad-hod Ranking with Kernel Pooling Chunk of text: Instead, it customizes word embeddings for search tasks. Nalisnick et al. propose to match query and documents using both the input and output of the embedding model, instead of only using one side of them . Diaz et al. nd that word embeddings trained locally on pseudo relevance feedback documents are more related to the query’s information needs, and can provide beer query expansion terms . Current neural ranking models fall into two groups: representation based and interaction based . e earlier focus of neural IR was mainly on representation based models, in which the query and documents are rst embedded into continuous vectors, and the ranking is calculated from their embeddings’ similarity. For example, DSSM and its convolutional version CDSSM map words to leer-tri-grams, embed query and documents using neural networks built upon the leer-tri-grams, and rank documents using their embedding similarity with the query.

[REF6] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF25\_d51ed05fd05b9d222427a05a87ed88217447b44f.pdf Title: A Deep Relevance Matching Model for Ad-hoc Retrieval Chunk of text: For example, MV-LSTM used K-max pooling strategy to select the top K strongest interaction signals from the matching matrix as the input of a MLP. However, such a pooling strategy simply truncates the signals and thus will be strongly biased to long documents since it is more likely for long documents to contain more strong signals. The pooling strategy is applied over the entire matching matrix in MV-LSTM, making it possible that the top K strongest signals all come from the interactions between a single query term and the document terms. In contrast, our model does not rely on any pooling strategy to truncate the interactions so that we can avoid these problems. Term Gating Network: One significant difference of our model from existing interaction-focused models is that we employ a joint deep architecture at the query term level.

[REF7] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF25\_d51ed05fd05b9d222427a05a87ed88217447b44f.pdf Title: A Deep Relevance Matching Model for Ad-hoc Retrieval Chunk of text: On the other hand, deep neural networks, as a representation learning method, are able to discover from the training data the hidden structures and features at different levels of abstraction that are useful for the tasks. Recently, deep models have been applied to a variety of applications in computer vision , speech recognition and NLP [25, 17], and have yielded significant performance improvements. Given the success of deep learning in these domains, it seems that deep learning should have a major impact on IR. However, there have been few positive results of deep models on IR tasks, especially ad-hoc retrieval tasks, until now. Without loss of generality, when applying deep models to ad-hoc retrieval, the task is typically formalized as a matching problem between two pieces of text (i.e., the query and document).

[REF8] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF25\_d51ed05fd05b9d222427a05a87ed88217447b44f.pdf Title: A Deep Relevance Matching Model for Ad-hoc Retrieval Chunk of text: Based on these differences, we propose a deep relevanceFigure 1: Two types of deep matching models: (a) Representation-focused models employ a Siamese (symmetric) architecture over the text inputs; (b) Interaction-focused models employ a hierarchical deep architecture over the local interaction matrix. matching model (DRMM) for ad-hoc retrieval by explicitly modeling the three major factors in relevance matching. Overall, our model is an interaction-focused model which employs a joint deep architecture at the query term1 level for relevance matching. Specifically, we first build local interactions between each pair of terms from a query and a document based on term embeddings. For each query term, we map the variable-length local interactions into a fixed-length matching histogram. Based on this fixed-length matching histogram, we then employ a feed forward matching network to learn hierarchical matching patterns and produce a matching score. Finally, the overall matching score is generated by aggregating the scores from each query term with a term gating network computing the aggregation weights.

[REF9] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Convolutional\_Neural\_Networks/BIBREF25\_d51ed05fd05b9d222427a05a87ed88217447b44f.pdf Title: A Deep Relevance Matching Model for Ad-hoc Retrieval Chunk of text: , Φ maps each text to a sequence of word vectors,while F is a CNN over the interaction matrix between word vectors from the two texts. Without loss of generality, all the model architectures of interaction-focused models can be viewed as a hierarchical deep architecture over the local interaction matrix, as shown in Figure 1(b). Although various deep matching models have been proposed under such a general matching problem formalization, most of them have only been demonstrated to be effective on a set of NLP tasks such as paraphrase identification and QA [11, 26]. There have been few positive results on the ad-hoc retrieval task. Even the deep models specially designed for Web search, e.g., DSSM and C-DSSM, were only evaluated on <query, doc title> pairs which are not a typical ad-hoc retrieval setting. If we directly apply these deep matching models on some benchmark retrieval collections, e.g. TREC collections, we find relatively poor performance compared to traditional ranking models, such as the language model and BM25 .

........................................................................................................................................................................................................

Title: Interaction-focused Systems - Pre-trained Language Models

Pre-trained language models have emerged as a powerful tool in neural information retrieval, enabling significant improvements in various natural language processing tasks [REF4]. In recent years, there has been a growing interest in leveraging pre-trained language models for interaction-focused systems, which aim to enhance the retrieval process by incorporating user interactions and feedback. One prominent approach in this domain is the use of pre-trained language models, such as BERT (Bidirectional Encoder Representations from Transformers) [REF0].

Pre-training of language models involves training them on large corpora of text data to learn general language representations [REF4]. For instance, BERT is pre-trained on the BooksCorpus and English Wikipedia, which provide a diverse range of textual information [REF0]. The use of document-level corpora, as opposed to shuffled sentence-level corpora, allows for the extraction of long contiguous sequences, which is critical for capturing contextual information [REF0].

Fine-tuning BERT for interaction-focused systems is relatively straightforward due to its self-attention mechanism and flexibility in modeling downstream tasks [REF0]. By swapping out the appropriate inputs and outputs, BERT can effectively encode text pairs and apply bidirectional cross attention [REF0]. This approach has been successfully applied in various applications involving text pairs, such as natural language inference and question answering [REF0].

The choice of pre-training data sets plays a crucial role in the performance of pre-trained language models [REF1]. It has been observed that pre-training on in-domain unlabeled data can significantly improve performance on downstream tasks [REF1]. For example, pre-training BERT on text from research papers improved its performance on scientific tasks [REF1]. Additionally, pre-training on a more diverse data set has been shown to yield improvements on downstream tasks [REF3]. Therefore, the selection of appropriate pre-training data sets, considering both domain relevance and diversity, is essential for achieving optimal performance [REF1][REF3].

To ensure computational efficiency during unsupervised pre-training, it is recommended to use objectives that produce short target sequences [REF2]. This allows for more efficient training and better utilization of computational resources [REF2]. Furthermore, the size and diversity of the pre-training data set are crucial factors in achieving high performance [REF7]. Larger and more diverse data sets, such as the Colossal Clean Crawled Corpus (C4), have been shown to enhance generic language understanding tasks [REF2][REF7].

In conclusion, pre-trained language models, particularly BERT, have shown great promise in interaction-focused systems for neural information retrieval. The choice of pre-training data sets, the use of appropriate objectives, and the consideration of data size and diversity are critical factors in achieving optimal performance. By leveraging pre-trained language models, researchers and practitioners can enhance the retrieval process and improve the overall effectiveness of information retrieval systems.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF21\_df2b0e26d0599ce3e70df8a9da02e51594e0e992.pdf Title: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding Chunk of text: Pre-training data The pre-training procedure largely follows the existing literature on language model pre-training. For the pre-training corpus we use the BooksCorpus (800M words) (Zhu et al., 2015) and English Wikipedia (2,500M words). For Wikipedia we extract only the text passages and ignore lists, tables, and headers. It is critical to use a document-level corpus rather than a shuffled sentence-level corpus such as the Billion Word Benchmark (Chelba et al., 2013) in order to extract long contiguous sequences. 3.2 Fine-tuning BERT Fine-tuning is straightforward since the selfattention mechanism in the Transformer allows BERT to model many downstream tasks— whether they involve single text or text pairs—by swapping out the appropriate inputs and outputs. For applications involving text pairs, a common pattern is to independently encode text pairs before applying bidirectional cross attention, such as Parikh et al. (2016); Seo et al. (2017).

[REF1] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF36\_3cfb319689f06bf04c2e28399361f414ca32c4b3.pdf Title: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Chunk of text: Similarly, using the RealNews-like data set for pre-training conferred an increase from 68.16 to 73.72 on the Exact Match score for ReCoRD, a data set that measures reading comprehension on news articles. As a final example, using data from Wikipedia produced significant (but less dramatic) gains on SQuAD, which is a question-answering data set with passages sourced from Wikipedia. Similar observations have been made in prior work, e.g. Beltagy et al. (2019) found that pre-training BERT on text from research papers improved its performance on scientific tasks. The main lesson behind these findings is that pre-training on in-domain unlabeled data can improve performance on downstream tasks. This is unsurprising but also unsatisfying if our goal is to pre-train a model that can rapidly adapt to language tasks from arbitrary domains. Liu et al. (2019c) also observed that pre-training on a more diverse data set yielded improvements on downstream tasks.

[REF2] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF36\_3cfb319689f06bf04c2e28399361f414ca32c4b3.pdf Title: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Chunk of text: As a result, we suggest using objectives that produce short target sequences so that unsupervised pre-training is more computationally efficient. Data sets We introduced the “Colossal Clean Crawled Corpus” (C4), which comprises heuristically-cleaned text from the Common Crawl web dump. When comparing C4 to 41Raffel, Shazeer, Roberts, Lee, Narang, Matena, Zhou, Li and Liu data sets that use additional filtering, we found that training on in-domain unlabeled data could boost performance in a few downstream tasks. However, constraining to a single domain typically results in a smaller data set. We separately showed that performance can degrade when an unlabeled data set is small enough that it is repeated many times over the course of pre-training. This motivates the use of a large and diverse data set like C4 for generic language understanding tasks. Training strategies We found that the basic approach of updating all of a pre-trained model’s parameters during fine-tuning outperformed methods that are designed to update fewer parameters, although updating all parameters is most expensive.

[REF3] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF36\_3cfb319689f06bf04c2e28399361f414ca32c4b3.pdf Title: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Chunk of text: This is unsurprising but also unsatisfying if our goal is to pre-train a model that can rapidly adapt to language tasks from arbitrary domains. Liu et al. (2019c) also observed that pre-training on a more diverse data set yielded improvements on downstream tasks. This observation also motivates the parallel line of research on domain adaptation for natural language processing; for surveys of this field see e.g. Ruder (2019); Li (2012). A drawback to only pre-training on a single domain is that the resulting data sets are often substantially smaller. Similarly, while the WebText-like variant performed as well or better than the C4 data set in our baseline setting, the Reddit-based filtering produced a data set that was about 40× smaller than C4 despite being based on 12× more data from Common Crawl. Note, however, that in our baseline setup we only pre-train on 2 35 ≈ 34B tokens, which is only about 8 times larger than the smallest pre-training data set we consider. We investigate at what point using a smaller pre-training data sets poses an issue in the following section.

[REF4] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF21\_df2b0e26d0599ce3e70df8a9da02e51594e0e992.pdf Title: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding Chunk of text: Introduction Language model pre-training has been shown to be effective for improving many natural language processing tasks (Dai and Le, 2015; Peters et al., 2018a; Radford et al., 2018; Howard and Ruder, 2018). These include sentence-level tasks such as natural language inference (Bowman et al., 2015; Williams et al., 2018) and paraphrasing (Dolan and Brockett, 2005), which aim to predict the relationships between sentences by analyzing them holistically, as well as token-level tasks such as named entity recognition and question answering, where models are required to produce fine-grained output at the token level (Tjong Kim Sang and De Meulder, 2003; Rajpurkar et al., 2016). There are two existing strategies for applying pre-trained language representations to downstream tasks: feature-based and fine-tuning. The feature-based approach, such as ELMo (Peters et al., 2018a), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018), introduces minimal task-specific parameters, and is trained on the downstream tasks by simply fine-tuning all pretrained parameters. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations. We argue that current techniques restrict the power of the pre-trained representations, especially for the fine-tuning approaches.

[REF5] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF22\_077f8329a7b6fa3b7c877a57b81eb6c18b5f87de.pdf Title: RoBERTa: A Robustly Optimized BERT Pretraining Approach Chunk of text: WNLI: We found the provided NLI-format data to be challenging to work with. Instead we use the reformatted WNLI data from SuperGLUE (Wang et al., 2019a), which indicates the span of the query pronoun and referent. We finetune RoBERTa using the margin ranking loss from Kocijan et al. (2019). For a given input sentence, we use spaCy (Honnibal and Montani, 2017) to extract additional candidate noun phrases from the sentence and finetune our model so that it assigns higher scores to positive referent phrases than for any of the generated negative candidate phrases. One unfortunate consequence of this formulation is that we can only make use of the positive training examples, which excludes over half of the provided training examples.10 10While we only use the provided WNLI training data, our Results We present our results in Table 5. In the first setting (single-task, dev), RoBERTa achieves state-of-the-art results on all 9 of the GLUE task development sets.

[REF6] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF36\_3cfb319689f06bf04c2e28399361f414ca32c4b3.pdf Title: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Chunk of text: 3.2.3 Objectives As an unsupervised objective, we will consider both a basic language modeling objective as well as our baseline denoising objective described in Section 3.1.4. We include the language modeling objective due to its historic use as a pre-training objective (Dai and Le, 2015; Ramachandran et al., 2016; Howard and Ruder, 2018; Radford et al., 2018; Peters et al., 2018) as well as its natural fit for the language model architectures we consider. For models that ingest a prefix before making predictions (the encoder-decoder model and prefix LM), we sample a span of text from our unlabeled data set and choose a random point to split it into prefix and target portions. For the standard language model, we train the model to predict the entire span from beginning to end. Our unsupervised denoising objective is designed for text-to-text models; to adapt it for use with a language model we concatenate the inputs and targets as described in Section 3.2.1. 3.2.4 Results The scores achieved by each of the architectures we compare are shown in Table 2. For all tasks, the encoder-decoder architecture with the denoising objective performed best.

[REF7] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF36\_3cfb319689f06bf04c2e28399361f414ca32c4b3.pdf Title: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Chunk of text: However, unlike objectives and benchmarks, new pre-training data sets are usually not treated as significant contributions on their own and are often not released alongside pre-trained models and code. Instead, they are typically introduced in the course of presenting a new method or model. As a result, there has been relatively little comparison of different pre-training data sets as well as a lack of a “standard” data set used for pre-training. Some recent notable exceptions (Baevski et al., 2019; Liu et al., 2019c; Yang et al., 2019) have compared pre-training on a new large (often Common Crawl-sourced) data set to using a smaller preexisting data set (often Wikipedia). To probe more deeply into the impact of the pre-training data set on performance, in this section we compare variants of our C4 data set and other potential sources of pre-training data. We release all of the C4 data set variants we consider as part of TensorFlow Datasets.11 3.4.1 Unlabeled Data Sets In creating C4, we developed various heuristics to filter the web-extracted text from Common Crawl (see Section 2.2 for a description). We are interested in measuring whether this filtering results in improved performance on downstream tasks, in addition to comparing it to other filtering approaches and common pre-training data sets.

[REF8] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF22\_077f8329a7b6fa3b7c877a57b81eb6c18b5f87de.pdf Title: RoBERTa: A Robustly Optimized BERT Pretraining Approach Chunk of text: + pretrain longer 160GB 8K 300K 94.4/88.7 90.0 96.1 + pretrain even longer 160GB 8K 500K 94.6/89.4 90.2 96.4 BERTLARGE with BOOKS + WIKI 13GB 256 1M 90.9/81.8 86.6 93.7 XLNetLARGE with BOOKS + WIKI 13GB 256 1M 94.0/87.8 88.4 94.4 + additional data 126GB 2K 500K 94.5/88.8 89.8 95.6 Table 4: Development set results for RoBERTa as we pretrain over more data (16GB → 160GB of text) and pretrain for longer (100K → 300K → 500K steps). Each row accumulates improvements from the rows above. RoBERTa matches the architecture and training objective of BERTLARGE . Results for BERTLARGE and XLNetLARGE are from Devlin et al. (2019) and Yang et al. (2019), respectively. Complete results on all GLUE tasks can be found in the Appendix.

[REF9] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Pre-trained\_Language\_Models/BIBREF22\_077f8329a7b6fa3b7c877a57b81eb6c18b5f87de.pdf Title: RoBERTa: A Robustly Optimized BERT Pretraining Approach Chunk of text: When controlling for training data, we observe that RoBERTa provides a large improvement over the originally reported BERTLARGE results, reaffirming the importance of the design choices we explored in Section 4. Next, we combine this data with the three additional datasets described in Section 3.2. We train RoBERTa over the combined data with the same number of training steps as before (100K). In total, we pretrain over 160GB of text. We observe further improvements in performance across all downstream tasks, validating the importance of data size and diversity in pretraining.9 Finally, we pretrain RoBERTa for significantly longer, increasing the number of pretraining steps from 100K to 300K, and then further to 500K. We again observe significant gains in downstream task performance, and the 300K and 500K step models outperform XLNetLARGE across most tasks. We note that even our longest-trained model does not appear to overfit our data and would likely benefit from additional training.

........................................................................................................................................................................................................

Title: Interaction-focused Systems - Ranking with Encoder-only Models

In recent years, there has been significant progress in designing ranking architectures for effectively scoring query-document pairs [REF0]. At the same time, pretrained contextualized language models, such as ELMo and BERT, have shown great promise in various natural language processing tasks [REF0]. These models leverage pre-training on large corpora and minimal task fine-tuning to capture contextual information [REF0]. Incorporating contextual information has been suggested to be valuable for ranking tasks [REF0]. For instance, ConvKNRM, a neural ranking model, utilizes a convolutional neural network to learn representations that are aware of context in local proximity [REF0].

Traditional ranking models heavily rely on manual feature engineering, which can be time-consuming and limited in capturing contextual information [REF1]. In contrast, neural ranking models offer the potential to obviate the need for handcrafted features by leveraging continuous vector space representations and neural architectures [REF1]. Notable neural ranking models include DRMM, DUET, KNRM, and Co-PACRR [REF1]. However, it is important to note that most neural ranking models, including the aforementioned ones, are re-ranking models that operate over a list of candidate documents [REF1]. This highlights the need for multi-stage ranking approaches that can effectively leverage contextual information.

Pretrained contextualized language models, such as BERT, have been successfully applied to search-related tasks, including document ranking [REF2]. Building on this, recent work has proposed multi-stage ranking architectures that incorporate BERT variants, such as monoBERT and duoBERT [REF2]. These models treat document ranking as a binary or pairwise classification problem and are arranged as stages in a pipeline, balancing the candidate set size with model complexity [REF2]. This approach allows for the benefits of richer models while controlling inference latencies [REF2].

Contextualized language models capture contextual information by incorporating multiple stacked layers of representations [REF3]. Deeper layers are believed to incorporate more context, and thus, it is beneficial to incorporate the output of all layers into the model [REF3]. This results in a three-dimensional similarity representation that expands the similarity representation conditioned on the query and document context [REF3]. By incorporating contextualized language models into existing neural ranking architectures, such as PACRR, KNRM, and DRMM, researchers have observed improved ad-hoc document ranking performance [REF4]. This approach involves using multiple similarity matrices, one for each layer of the language model, and has been shown to achieve state-of-the-art performance on various datasets [REF4].

In summary, interaction-focused systems that focus on ranking with encoder-only models have gained significant attention in the field of neural information retrieval. These systems leverage pretrained contextualized language models, such as BERT, to capture contextual information and improve ranking performance. Multi-stage ranking architectures, such as monoBERT and duoBERT, have been proposed to balance the benefits of richer models with inference latencies. Additionally, incorporating contextualized language models into existing neural ranking architectures has shown promising results in improving ad-hoc document ranking performance.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: INTRODUCTION Recently, there has been much work designing ranking architectures to effectively score query-document pairs, with encouraging results [5, 6, 20]. Meanwhile, pretrained contextualized language models (such as ELMo and BERT ) have shown great promise on various natural language processing tasks [4, 16]. These models work by pre-training LSTM-based or transformer-based language models on a large corpus, and then by performing minimal task fine-tuning (akin to ImageNet [3, 23]). Prior work has suggested that contextual information can be valuable when ranking. ConvKNRM , a recent neural ranking model, uses a convolutional neural network atop word representations, allowing the model to learn representations aware of context in local proximity.

[REF1] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF40\_63a2fabbe4b1615a84d5f4d90987733cf09e3ff8.pdf Title: Multi-Stage Document Ranking with BERT Chunk of text: In our work, we make the connection between BERTbased models and multi-stage ranking, which allows us to trade off the quality of the results with inference latency. The advent of deep learning has brought tremendous excitement into the information retrieval community. Although machine-learned ranking models have been well studied since the mid-2000s under the banner of “learning to rank”, the paradigm is heavily driven by manual feature engineering (Liu, 2009; Li, 2011); commercial web search engines are known to incorporate thousands of features (or more) in their models. Continuous vector space representations coupled with neural models promise to obviate the need for handcrafted features and have attracted the attention of many researchers. Well-known neural ranking models include DRMM (Guo et al., 2016), DUET (Mitra et al., 2017), KNRM (Xiong et al., 2017), and Co-PACRR (Hui et al., 2018); the literature is too vast for an exhaustive review here, and thus we refer readers to recent overviews (Onal et al., 2018; Mitra and Craswell, 2019). Although often glossed over, most neural ranking models today (including all the models referenced above) are actually re-ranking models, in the sense that they operate over the output of a list of candidate documents, typically produced by a “bag of words” query.

[REF2] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF40\_63a2fabbe4b1615a84d5f4d90987733cf09e3ff8.pdf Title: Multi-Stage Document Ranking with BERT Chunk of text: Introduction Neural models pre-trained on language modeling tasks such as ELMo (Peters et al., 2017), OpenAI GPT (Radford et al., 2018), and BERT (Devlin et al., 2019) have achieved impressive results on NLP tasks ranging from natural language inference to question answering. One such popular model, BERT, has recently been applied to search-related tasks, retrieval-based question answering (Yang et al., 2019b), as well as document ranking (Yang et al., 2019c; MacAvaney et al., 2019; Yilmaz et al., 2019). This paper builds on previous initial work (Nogueira and Cho, 2019) to tackle the document ranking problem with a multi-stage ranking architecture. We introduce two BERT variants, called monoBERT and duoBERT. The monoBERT model treats document ranking as a binary classification problem over individual candidate documents, while the duoBERT model adopts a pairwise classification approach that considers pairs of candidate documents. For end-to-end document ranking, we arrange these models as stages in a pipeline where each balances the size of the candidate set against the inherent complexity of the model. This design allows us to obtain the benefits of richer models while controlling the increased inference latencies that come with these richer models.

[REF3] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: For example, the contextualized representation of word bank would be different in bank deposit and river bank, while a pretrained word embedding model would always result in the same representation for this term. Given that these representations capture contextual information in the language, we investigate how these models can also benefit general neural ranking models. Although contextualized language models vary in particular architectures, they typically consist of multiple stacked layers of representations (e.g., recurrent or transformer outputs). The intuition is that the deeper the layer, the more context is incorporated. To allow neural ranking models to learn which levels are most important, we choose to incorporate the output of all layers into the model, resulting in a three-dimensional similarity representation. Thus, we expand the similarity representation (conditioned on the query and document context) to SQ,D ∈ R L× |Q |× |D | where L is the number of layers in the model, akin to the channel in image processing. Let contextQ,D(t,l)

[REF4] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: $15.00 <https://doi.org/10.1145/3331184.3331317> an approach that learns a recurrent neural network for term representations, thus being able to capture context from the entire text . These approaches are inherently limited by the variability found in the training data. Since obtaining massive amounts of highquality relevance information can be difficult , we hypothesize that pretrained contextualized term representations will improve ad-hoc document ranking performance. We propose incorporating contextualized language models into existing neural ranking architectures by using multiple similarity matrices – one for each layer of the language model. We find that, at the expense of computation costs, this improves ranking performance considerably, achieving state-of-the-art performance on the Robust 2004 and WebTrack 2012–2014 datasets. We also show that combining each model with BERT’s classification mechanism can further improve ranking performance. We call this approach CEDR (Contextualzed Embeddings for Document Ranking).

[REF5] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: In summary, our contributions are as follows: - We are the first to demonstrate that contextualized word representations can be successfully incorporated into existing neural architectures (PACRR , KNRM , and DRMM ), allowing them to leverage contextual information to improve ad-hoc document ranking. - We present a new joint model that combines BERT’s classification vector with existing neural ranking architectures (using BERT’s token vectors) to get the benefits from both approaches. - We demonstrate an approach for addressing the performance impact of computing contextualized language models by only partially computing the language model representations. - Our code is available for replication and future work.1 2 METHODOLOGY 2.1 Notation In ad-hoc ranking, documents are ranked for a given query according to a relevance estimate.

[REF6] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF40\_63a2fabbe4b1615a84d5f4d90987733cf09e3ff8.pdf Title: Multi-Stage Document Ranking with BERT Chunk of text: Well-known neural ranking models include DRMM (Guo et al., 2016), DUET (Mitra et al., 2017), KNRM (Xiong et al., 2017), and Co-PACRR (Hui et al., 2018); the literature is too vast for an exhaustive review here, and thus we refer readers to recent overviews (Onal et al., 2018; Mitra and Craswell, 2019). Although often glossed over, most neural ranking models today (including all the models referenced above) are actually re-ranking models, in the sense that they operate over the output of a list of candidate documents, typically produced by a “bag of words” query. Thus, document retrieval with neural models today already uses multi-stage ranking, albeit an impoverished form with only a single re-ranking stage. This recognition provides a starting point of our work, from which we build BERT-based multi-stage ranking. 3 Multi-Stage Ranking with BERT In our formulation, a multi-stage ranking architecture comprises a number of stages, denoted H0 to HN . Except for H0, which retrieves k0 candidates from an inverted index, each stage Hn receives a ranked list Rn−1 comprising kn−1 candidates from the previous stage. Each stage, in turn, provides a ranked list Rn comprising kn candidates to the subsequent stage, with the obvious requirement that kn ≤ kn−1.

[REF7] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: It shows that most of the performance benefits can be achieved by only running BERT through layer 5; the performance is comparable to running the full BERT model, while running more than twice as fast. While we acknowledge that our research code is not completely optimized, we argue that this approach is generally applicable because the processing of these layers are sequential, query-dependent, and dominate the processing time of the entire model. This approach is a simple time-saving measure. 4 CONCLUSION We demonstrated that contextualized word embeddings can be effectively incorporated into existing neural ranking architectures and suggested an approach for improving runtime performance by limiting the number of layers processed.

[REF8] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF40\_63a2fabbe4b1615a84d5f4d90987733cf09e3ff8.pdf Title: Multi-Stage Document Ranking with BERT Chunk of text: As demonstrated by Lin (2019), in a limited data regime, it is not entirely clear that neural techniques actually perform better than welltuned “classic” IR techniques; subsequent work by Yang et al. (2019a) show that the gains are modest at best. Until recently, research in neural ranking models mostly took advantage of proprietary datasets derived from user behavior logs (which large organizations can gather in abundance). Since these datasets cannot be shared,only a small set of researchers could productively work on neural ranking models and different models could not be easily compared; the combination of both factors hamper rapid progress. Fortunately, the field has seen the release of two large-scale datasets for powering data-hungry neural models: MS MARCO (Bajaj et al., 2018) and TREC CAR (Dietz et al., 2017). We take advantage of both datasets to train our models, which we detail below. 4.1 MS MARCO The Microsoft MAchine Reading COmprehension dataset (MS MARCO) is a large-scale resource created from approximately half a million anonymized questions sampled from Bing’s search query logs.

[REF9] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-only\_Models/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: [cs.IR] 19 Aug 2019a real-valued relevance estimate for the document to the query. Neural relevance ranking architectures generally use a similarity matrix as input S ∈ R |Q |× |D | , where each cell represents a similarity score between the query and document: Si,j = sim(qi ,dj). These similarity values are usually the cosine similarity score between the word vectors of each term in the query and document. 2.2 Contextualized similarity tensors Pretrained contextual language representations (such as those from ELMo and BERT ) are context sensitive; in contrast to more conventional pretrained word vectors (e.g., GloVe ) that generate a single word representation for each word in the vocabulary, these models generate a representation of each word based on its context in the sentence. For example, the contextualized representation of word bank would be different in bank deposit and river bank, while a pretrained word embedding model would always result in the same representation for this term. Given that these representations capture contextual information in the language, we investigate how these models can also benefit general neural ranking models.

........................................................................................................................................................................................................

Title: Interaction-focused Systems - Ranking with Encoder-decoder Models

In the field of Neural Information Retrieval, interaction-focused systems have gained significant attention due to their ability to improve the ranking of documents based on user interactions. One popular approach in this area is the use of encoder-decoder models, particularly those pretrained with language modeling objectives such as BERT [REF0]. These models have shown remarkable effectiveness in various classification and sequence labeling tasks in Natural Language Processing (NLP) [REF0]. Nogueira and Cho were the first to demonstrate the effectiveness of BERT in ranking tasks, specifically in document ranking for information retrieval [REF0].

In the context of document ranking, the simplest formulation is to deploy a classifier that estimates the probability of each document belonging to the "relevant" class and then sort all the candidates based on these estimates [REF0]. However, applying inference to every document in a large corpus with respect to a query is impractical. Therefore, these techniques are typically applied to rerank a list of candidates obtained from a keyword search using a traditional Information Retrieval (IR) scoring function like BM25 [REF0] [REF1].

The standard multi-stage pipeline architecture for interaction-focused systems involves keyword retrieval followed by reranking using one or more machine learning models [REF1]. In this architecture, the candidates for reranking are obtained from the results of a keyword search using a classic IR scoring function like BM25 [REF1]. This approach has been widely used and serves as a baseline for comparison in many studies [REF2] [REF3].

To adapt encoder-decoder models to the task of document reranking, recent research has explored the use of pretrained sequence-to-sequence models such as T5 [REF1]. This novel use of sequence-to-sequence models for document reranking has shown promising results, particularly in data-poor regimes where limited training examples are available [REF1] [REF5]. The sequence-to-sequence models, like T5, leverage their latent knowledge and semantic understanding to improve the reranking process [REF5].

In the evaluation of interaction-focused systems, various baselines are commonly used for comparison. These include BM25, BM25+RM3 (query expansion), and BM25+BERT-large (two-stage pipeline with BM25 and BERT reranker) [REF2] [REF3]. These baselines provide a fair yet competitive comparison point for assessing the performance of encoder-decoder models in document reranking [REF3].

In terms of datasets, the MS MARCO dataset is often used for evaluating interaction-focused systems [REF4]. This dataset consists of queries and relevance judgments on a collection of documents, and it is commonly used as a held-out test set in zero-shot transfer settings [REF4]. The evaluation metrics for these systems typically include rank-based metrics such as mean precision at k (P@k) [REF7].

In conclusion, interaction-focused systems that employ encoder-decoder models, such as pretrained sequence-to-sequence models, have shown promise in improving document ranking in Neural Information Retrieval. These models leverage their latent knowledge and semantic understanding to enhance the reranking process. Baseline models and evaluation metrics play a crucial role in assessing the performance of these systems.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF43\_f6e0164466e827112fd415afdc28ddf8e0eb1ba3.pdf Title: Document Ranking with a Pretrained Sequence-to-Sequence Model Chunk of text: Introduction A simple, straightforward formulation of ranking is to convert the task into a classification problem, and then sort the candidate items to be ranked based on the probability that each item belongs to the desired class. Applied to the document ranking problem in information retrieval—where given a query, the system’s task is to return a ranked list of documents from a large corpus that maximizes some ranking metric such as average precision or nDCG—the simplest formulation is to deploy a classifier that estimates the probability each document belongs to the “relevant” class, and then sort all the candidates by these estimates. Deep transformer models pretrained with language modeling objectives, exemplified by BERT , have proven highly effective in a variety of classification and sequence labeling tasks in NLP. Nogueira and Cho were the first to demonstrate its effectiveness in ranking tasks. Since it is impractical to apply inference to every document in a corpus with respect to a query, these techniques are typically applied to rerank a list of candidates. In a typical end-to-end system, these candidates are from the results of a keyword search based on a “classic” IR scoring function such as BM25 . This gives rise to the standard multi-stage pipeline architecture of keyword retrieval followed by reranking using one or more machine learning models [1, 10].

[REF1] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF43\_f6e0164466e827112fd415afdc28ddf8e0eb1ba3.pdf Title: Document Ranking with a Pretrained Sequence-to-Sequence Model Chunk of text: In a typical end-to-end system, these candidates are from the results of a keyword search based on a “classic” IR scoring function such as BM25 . This gives rise to the standard multi-stage pipeline architecture of keyword retrieval followed by reranking using one or more machine learning models [1, 10]. The contribution of this work is to adapt a pretrained sequence-to-sequence model (in our case, T5 ) to the task of document reranking. To our knowledge, this is a novel use of this class of models that has not been previously described in the literature. In a data-rich regime, with lots of training examples, our method can outperform a pure classification-based encoder-only approach. ∗Equal contribution. However, the sequence-to-sequence model appears to be far more data-efficient: our approach shines in a data-poor regime and significantly outperforms BERT with limited training examples.

[REF2] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF43\_f6e0164466e827112fd415afdc28ddf8e0eb1ba3.pdf Title: Document Ranking with a Pretrained Sequence-to-Sequence Model Chunk of text: We select the highest probability among these passages as the relevance probability of the document. 3.3 Baselines We compare our method against the following baselines: BM25: For baseline bag-of-words retrieval, we use the BM25 implementation in the Anserini opensource IR toolkit ,2 which is based on Lucene. We adopt all the default settings. At inference time, we retrieve the top 1000 documents per query. BM25+RM3: To examine the effects of query expansion, we applied the BM25+RM3 model as described in Yang et al. , where it is shown to be a competitive baseline for (pre-BERT) neural ranking models. We use the implementation in Anserini, with all default settings.

[REF3] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF43\_f6e0164466e827112fd415afdc28ddf8e0eb1ba3.pdf Title: Document Ranking with a Pretrained Sequence-to-Sequence Model Chunk of text: , where it is shown to be a competitive baseline for (pre-BERT) neural ranking models. We use the implementation in Anserini, with all default settings. BM25+BERT-large: We additionally compare our method against the BERT-large condition from Nogueira et al. , which is a two-stage pipeline with bag-of-words retrieval (BM25) followed by a BERT reranker. Architecturally, it is the same as our method, the only difference being BERT vs. T5 as the reranking model. Nogueira et al. can be characterized as the baseline of the best methods from the official MS MARCO passage leaderboard; all higher-ranked submissions can be described as improvements upon this basic approach, and thus it represents a fair yet competitive comparison point.

[REF4] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF43\_f6e0164466e827112fd415afdc28ddf8e0eb1ba3.pdf Title: Document Ranking with a Pretrained Sequence-to-Sequence Model Chunk of text: It comprises 250 queries, with relevance judgments on a collection of 528K documents (TREC Disks 4 and 5), whose average length is 2,800 characters or 460 words. We use the topic “titles” (short keyword phrases, much like the input to a search engine) as queries to our bag-of-words retrieval methods (see Section 3.3) and the topic “descriptions” (sentence-length statements of information needs) as input to our sequence-to-sequence models. These topic descriptions are more similar to MS MARCO’s natural language questions, and others have found that using them improves the effectiveness of pre2trained reranking models . We do not train our models on this dataset, and use all its queries and relevance judgments as a held-out test set; thus, our evaluation adopts a zero-shot transfer setting. 3.2 Training and Inference We fine-tune our T5 models (base, large, and 3B) with a constant learning rate of 10−3 for 100k iterations with class-balanced batches of size 128. To simplify our training procedure (and related hyperparameters) as well as to eliminate the need for convergence checks, we simply trained for a fixed number of iterations, selected based on the computational demands of our largest model and the (self-allotted) time for running experiments. We report results using the model state at the final checkpoint.

[REF5] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF43\_f6e0164466e827112fd415afdc28ddf8e0eb1ba3.pdf Title: Document Ranking with a Pretrained Sequence-to-Sequence Model Chunk of text: ∗Equal contribution. However, the sequence-to-sequence model appears to be far more data-efficient: our approach shines in a data-poor regime and significantly outperforms BERT with limited training examples. The main advantage of our approach, we believe, is that by “connecting” fine-tuned latent representations of relevance to related output “target words”, we can exploit the model’s latent knowledge (e.g., of semantics, linguistic relations, etc.) that has been honed through pretraining. We describe probing experiments that attempt to verify our intuitions by deliberately altering the target words to capture different aspects of “semantic relatedness”. 2 Method Our reranking method is based on T5 , which is a sequence-to-sequence model that uses a similar masked language modeling objective as BERT to pretrain its encoder–decoder architecture. In this model, all target tasks are cast as sequence-to-sequence tasks. For our task, the input sequence is: Query: q Document:

[REF6] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF36\_3cfb319689f06bf04c2e28399361f414ca32c4b3.pdf Title: Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer Chunk of text: We explore the performance of different model architectures in Section 3.2. Our baseline model is designed so that the encoder and decoder are each similar in size and configuration to a “BERTBASE” (Devlin et al., 2018) stack. Specifically, both the encoder and decoder consist of 12 blocks (each block comprising self-attention, optional encoder-decoder attention, and a feed-forward network). The feed-forward networks in each block consist of a dense layer with an output dimensionality of dff = 3072 followed by a ReLU nonlinearity and another dense layer. The “key” and “value” matrices of all attention mechanisms have an inner dimensionality of dkv = 64 and all attention mechanisms have 12 heads. All other sub-layers and embeddings have a dimensionality of dmodel = 768. In total, this results in a model with about 220 million parameters.

[REF7] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF42\_d0086b86103a620a86bc918746df0aa642e2a8a3.pdf Title: Language Models as Knowledge Bases? Chunk of text: On the retrieved top k articles, a neural reading comprehension model then extracts answers. To avoid giving the language models a competitive advantage, we constrain the predictions of DrQA to single-token answers. 4.4 Metrics We consider rank-based metrics and compute results per relation along with mean values across all relations. To account for multiple valid objects for a subject-relation pair (i.e., for N-M relations), we follow Bordes et al. (2013) and remove from the candidates when ranking at test time all other valid objects in the training data other than the one we test. We use the mean precision at k (P@k). For a given fact, this value is 1 if the object is ranked among the top k results, and 0 otherwise.

[REF8] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF42\_d0086b86103a620a86bc918746df0aa642e2a8a3.pdf Title: Language Models as Knowledge Bases? Chunk of text: Freq: For a subject and relation pair, this baseline ranks words based on how frequently they appear as objects for the given relation in the test data. It indicates the upper bound performance of a model that always predicts the same objects for a particular relation. RE: For the relation-based knowledge sources, we consider the pretrained Relation Extraction (RE) model of Sorokin and Gurevych (2017). This model was trained on a subcorpus of Wikipedia annotated with Wikidata relations. It extracts relation triples from a given sentence using an LSTMbased encoder and an attention mechanism. Based on the alignment information from the knowledge sources, we provide the relation extractor with the sentences known to express the test facts.

[REF9] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Ranking\_with\_Encoder-decoder\_Models/BIBREF42\_d0086b86103a620a86bc918746df0aa642e2a8a3.pdf Title: Language Models as Knowledge Bases? Chunk of text: Our investigation reveals that (i) the largest BERT model from Devlin et al. (2018b) (BERT-large) captures (accurate) relational knowledge comparable to that of a knowledge base extracted with an off-the-shelf relation extractor and an oracle-based entity linker from a corpus known to express the relevant knowledge, (ii) factual knowledge can be recovered surprisingly well from pretrained language models, however, for some relations (particularly N-to-M relations) performance is very poor, (iii) BERT-large consistently outperforms other language models in recovering factual and commonsense knowledge while at the same time being more robust to the phrasing of a query, and (iv) BERT-large achieves remarkable results for open-domain QA, reaching 57.1% precision@10 compared to 63.5% of a knowledge base constructed using a task-specific supervised relation extraction system. 2 Background In this section we provide background on language models. Statistics for the models that we include in our investigation are summarized in Table 1. 2.1 Unidirectional Language Models Given an input sequence of tokens w = [w1,w2, . . . ,wN], unidirectional language models commonly assign a probability p(w) to the sequence by factorizing it as follows p(w) = Y t p(wt |wt−1, . . . ,w1). (1) A common way to estimate this probability is using neural language models (Mikolov and Zweig, 2012; Melis et al., 2017; Bengio et al., 2003) with p(wt |wt−1, . . .

........................................................................................................................................................................................................

Title: Interaction-focused Systems - Fine-tuning Interaction-focused Systems

Interaction-focused systems in neural information retrieval aim to improve the retrieval process by incorporating user interactions and feedback. One approach to enhancing these systems is through fine-tuning, where the system learns from user-provided information to optimize the retrieval performance [REF0].

The need for fine-tuning arises from the challenge of finding the "right" answer in information retrieval tasks, such as clustering or classification. Traditional algorithms often struggle to capture the user's subjective notion of similarity or meaningful clusters [REF1]. For instance, in clustering, different algorithms may produce different clusterings based on various criteria, such as authorship, topic, or writing style [REF1]. However, if the user desires a specific clustering based on a particular criterion, it is often difficult to convey this preference to the algorithm [REF1].

To address this issue, researchers have proposed methods that learn distance metrics to respect user-provided similarity relationships [REF2]. These methods go beyond focusing solely on the training set and instead learn a full metric over the input space, allowing for better generalization to unseen data [REF2]. By incorporating user feedback, these interaction-focused systems can improve the performance of unsupervised algorithms like K-means, which heavily rely on the quality of the metric [REF3].

One promising approach in interaction-focused systems is the use of similarity information to guide the clustering process [REF3]. Wagstaff et al. proposed a method that searches for a clustering solution that aligns with the user's notion of similarity [REF3]. However, similar to other methods like Multidimensional Scaling (MDS) and Locally Linear Embedding (LLE), these approaches often struggle to generalize to unseen data [REF3].

In addition to improving clustering performance, fine-tuning interaction-focused systems can also benefit other unsupervised algorithms and tasks [REF5]. For example, in the supervised learning setting, efforts have been made to define or learn local or global metrics for classification [REF5]. While these methods have shown success in classification tasks, their effectiveness in learning general metrics for algorithms like K-means remains uncertain, especially when dealing with less structured data [REF5].

Experimental results have demonstrated the effectiveness of fine-tuning interaction-focused systems. In various datasets, using a learned metric has led to significantly improved performance compared to naive K-means [REF4]. Furthermore, the combination of learned metrics with constrained K-means has shown even better results, outperforming constrained K-means alone [REF4]. However, the ease of learning metrics and the extent of improvement may vary depending on the problem and the amount of side-information available [REF6] [REF7].

In summary, fine-tuning interaction-focused systems through the learning of distance metrics can enhance the performance of information retrieval tasks. By incorporating user interactions and feedback, these systems can better capture the user's subjective notion of similarity and improve the quality of clustering and other unsupervised algorithms.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: One feature distinguishing our work from these is that we will learn a full metric over the input space, rather than focusing only on (finding an embedding for) the points in the training set. Our learned metric thus generalizes more easily to previously unseen data. More importantly, methods such as LLE and MDS also suffer from the “no right answer” problem: For example, if MDS finds an embedding that fails to capture the structure important to a user, it is unclear what systematic corrective actions would be available. (Similar comments also apply to Principal Components Analysis (PCA) .) As in our motivating clustering example, the methods we propose can also be used in a pre-processing step to help any of these unsupervised algorithms to find better solutions. In the supervised learning setting, for instance nearest neighbor classification, numerous attempts have been made to define or learn either local or global metrics for classification.

[REF1] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: Introduction The performance of many learning and datamining algorithms depend critically on their being given a good metric over the input space. For instance, K-means, nearest-neighbors classifiers and kernel algorithmssuch as SVMs all need to be given good metrics that reflect reasonably well the important relationships between the data. This problem is particularly acute in unsupervised settings such as clustering, and is related to the perennial problem of there often being no “right” answer for clustering: If three algorithms are used to cluster a set of documents, and one clusters according to the authorship, another clusters according to topic, and a third clusters according to writing style, who is to say which is the “right” answer? Worse, if an algorithm were to have clustered by topic, and if we instead wanted it to cluster by writing style, there are relatively few systematic mechanisms for us to convey this to a clustering algorithm, and we are often left tweaking distance metrics by hand. In this paper, we are interested in the following problem: Suppose a user indicates that certain points in an input space (say, ) are considered by them to be “similar.” Can we automatically learn a distance metric over that respects these relationships, i.e., one that assigns small distances between the similar pairs? For instance, in the documents example, we might hope that, by giving it pairs of documents judged to be written in similar styles, it would learn to recognize the critical features for determining style���������One important family of algorithms that (implicitly) learn metrics are the unsupervised ones that take an input dataset, and find an embedding of it in some space.

[REF2] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: Can we automatically learn a distance metric over that respects these relationships, i.e., one that assigns small distances between the similar pairs? For instance, in the documents example, we might hope that, by giving it pairs of documents judged to be written in similar styles, it would learn to recognize the critical features for determining style���������One important family of algorithms that (implicitly) learn metrics are the unsupervised ones that take an input dataset, and find an embedding of it in some space. This includes algorithms such as Multidimensional Scaling (MDS) , and Locally Linear Embedding (LLE) . One feature distinguishing our work from these is that we will learn a full metric over the input space, rather than focusing only on (finding an embedding for) the points in the training set. Our learned metric thus generalizes more easily to previously unseen data.

[REF3] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: While these methods often learn good metrics for classification, it is less clear whether they can be used to learn good, general metrics for other algorithms such as K-means, particularly if the information available is less structured than the traditional, homogeneous training sets expected by them. In the context of clustering, a promising approach was recently proposed by Wagstaff et al. for clustering with similarity information. If told that certain pairs are “similar” or “dissimilar,” they search for a clustering that puts the similar pairs into the same, and dissimilar pairs into different, clusters. This gives a way of using similarity side-information to find clusters that reflect a user’s notion of meaningful clusters. But similar to MDS and LLE, the (“instance-level”) constraints that they use do not generalize to previously unseen data whose similarity/dissimilarity to the training set is not known. We will later discuss this work in more detail, and also examine the effects of using the methods we propose in conjunction with these methods.

[REF4] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: In each, we ran one experiment using “little” side-information ­ , and one with “much” side-information. The results are given in Figure 6.9 We see that, in almost every problem, using a learned diagonal or full metric leads to significantly improved performance over naive K-means. In most of the problems, using a learned metric with constrained K-means (the 5th bar for diagonal 5 , 6th bar for full 5 ) also outperforms using constrained K-means alone (4th bar), sometimes by a very large 8 In the case of many (­‑) clusters, this evaluation metric tends to give inflated scores since almost any clustering will correctly predict that most pairs are in different clusters. In this setting, we therefore modified the measure averaging not only I J , IL drawn uniformly at random, but from the same cluster (as determined by ! ) with chance 0.5, and from different clusters with chance 0.5, so that “matches” and “mis-matches” are given the same weight. All results reported here used K-means with multiple restarts, and are averages over at least 20 trials (except for wine, 10 trials). 9F was generated by picking a random subset of all pairs of points sharing the same class !

[REF5] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: As in our motivating clustering example, the methods we propose can also be used in a pre-processing step to help any of these unsupervised algorithms to find better solutions. In the supervised learning setting, for instance nearest neighbor classification, numerous attempts have been made to define or learn either local or global metrics for classification. In these problems, a clear-cut, supervised criterion—classification error—is available and can be optimized for. (See also , for a different way of supervising clustering.) This literature is too wide to survey here, but some relevant examples include [10, 5, 3, 6], and also gives a good overview of some of this work. While these methods often learn good metrics for classification, it is less clear whether they can be used to learn good, general metrics for other algorithms such as K-means, particularly if the information available is less structured than the traditional, homogeneous training sets expected by them. In the context of clustering, a promising approach was recently proposed by Wagstaff et al.

[REF6] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: As shown by the accuracy scores given in the figure, both K-means and constrained K-means failed to find good clusterings. But by first learning a distance metric and then clustering according to that metric, we easily find the correct clustering separating the true clusters from each other. Figure 5 gives another example showing similar results. We also applied our methods to 9 datasets from the UC Irvine repository. Here, the “true clustering” is given by the data’s class labels. In each, we ran one experiment using “little” side-information ­ , and one with “much” side-information. The results are given in Figure 6.9 We see that, in almost every problem, using a learned diagonal or full metric leads to significantly improved performance over naive K-means.

[REF7] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: For some problems (e.g., wine), our algorithm learns good diagonal and full metrics quickly with only a very small amount of side-information; for some others (e.g., protein), the distance metric, particularly the full metric, appears harder to learn and provides less benefit over constrained K-means. 4 Conclusions We have presented an algorithm that, given examples of similar pairs of points in , learns a distance metric that respects these relationships. Our method is based on posing metric learning as a convex optimization problem, which allowed us to derive efficient, localoptima free algorithms. We also showed examples of diagonal and full metrics learned from simple artificial examples, and demonstrated on artificial and on UCI datasets how our methods can be used to improve clustering performance.

‑ ­ if and! are similar (1) How can we learn a distance metric "#! between points and # that respects this; specifically, so that “similar” points end up close to each other? Consider learning a distance metric of the form "#! %$ '&("#)$+\*,\* -.#/\*,\* & $+01-2#4365 7-.#98 (2) To ensure that this be a metric—satisfying non-negativity and the triangle inequality— we require that 5 be positive semi-definite, 5;:=<. 1 Setting 5>$@?

[REF9] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Fine-tuning\_Interaction-focused\_Systems/BIBREF44\_d1a2d203733208deda7427c8e20318334193d9d7.pdf Title: Distance metric learning, with application to clustering with side-information Chunk of text: Also, this problem has an objective that is linear in the parameters 5 , and both of the constraints are also easily verified to be convex. Thus, the optimization problem is convex, which enables us to derive efficient, local-minima-free algorithms to solve it. We also note that, while one might consider various alternatives to (4), “ & \*\* -!'\*,\* B& '­ ” would not be a good choice despite its giving a simple linear constraint. It would result in 5 always being rank 1 (i.e., the data are always projected onto a line).3 2.1 The case of diagonal 5 In the case that we want to learn a diagonal 5 $)(\*,+ 5C C5BCB 8 8 8ZC5 , we can derive an efficient algorithm using the Newton-Raphson method. Define - 5%$ - 5C 8 8 8Z5 )$ . \*,\* - \*,\* B& -0/1+324 .

........................................................................................................................................................................................................

Title: Interaction-focused Systems - Dealing with long texts

In the field of Neural Information Retrieval, interaction-focused systems have gained significant attention due to their ability to effectively handle long texts. Long texts, such as documents or passages, pose unique challenges in information retrieval tasks, as they contain a large amount of information that needs to be processed and understood. In this section, we will explore the use of interaction-focused systems in dealing with long texts, specifically focusing on the application of BERT (Bidirectional Encoder Representations from Transformers) [REF0] and its impact on ad-hoc document retrieval.

BERT, a state-of-the-art neural language model, has shown great potential in various natural language processing tasks, including information retrieval. It is trained to predict the relationship between two pieces of text, typically sentences, and its attention-based architecture models the local interactions of words in one text with words in another [REF0]. This interaction-based neural ranking model requires minimal search-specific architectural engineering, making it suitable for information retrieval tasks [REF0].

One approach to leveraging BERT for ad-hoc document retrieval is fine-tuning pre-trained BERT models with a limited amount of search data. Experimental studies have shown that this approach can achieve better performance than strong baselines [REF0] [REF7]. By adapting and fine-tuning BERT, the model can effectively capture the relevance patterns between queries and documents, leading to improved accuracy in search results [REF5] [REF6]. Furthermore, BERT's contextualized language modeling brings new possibilities to information retrieval, as it encodes general language patterns and enhances text understanding [REF4] [REF5].

In traditional retrieval models, shorter keyword queries have been the norm due to the limitations of bag-of-words approaches. However, with BERT, longer natural language queries have been found to outperform short keyword queries by large margins [REF7]. This observation highlights the importance of understanding natural language queries and the role of stopwords and punctuation in defining grammar structures and word dependencies [REF7]. By considering the entire context of a query, BERT can effectively extract key information from natural language, leading to more accurate search results [REF7].

Moreover, BERT can be enhanced with search knowledge from a large search log, enabling the model to have knowledge about both text understanding and the search task [REF7]. This combination of language modeling and search knowledge proves beneficial in related search tasks where labeled data are limited [REF5] [REF7]. By incorporating search signals and leveraging BERT's language understanding capabilities, the model can effectively handle long texts and improve the overall retrieval performance [REF0] [REF7].

In conclusion, interaction-focused systems, particularly those utilizing BERT, have shown promising results in dealing with long texts in information retrieval tasks. By leveraging BERT's language understanding capabilities and fine-tuning the model with search data, these systems can effectively capture relevance patterns, handle natural language queries, and improve retrieval performance. Further research in this area can explore additional techniques to enhance the interaction-focused systems and improve their effectiveness in dealing with long texts.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: BERT is a state-of-the-art neural language model. It also fits well with search tasks. BERT is trained to predict the relationship between two pieces of text (typically sentences); and its attention-based architecture models the local interactions of words in text1 with words in text2. It can be viewed as an interaction-based neural ranking model , thus minimal search-specific architectural engineering is required. This paper explores the effect of BERT’s language understanding on ad-hoc document retrieval. It examines BERT models on two adhoc retrieval datasets with different characteristics. Experiments show that fine-tuning pre-trained BERT models with a limited amount of search data can achieve better performance than strong baselines.

[REF1] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: 2 RELATED WORK Recent neural IR models have made promising progress in learning query-document relevance patterns. One line of research learns text presentations tailored for the search task [1, 2, 9] with search signals from click logs [1, 9] or pseudo-relevance feedback . Another line of research designs neural architectures to capture diverse matching features such as exact match signals and passage-level signals

[REF2] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: INTRODUCTION Text retrieval requires understanding document meanings and the search task. Neural networks are an attractive solution because they can acquire that understanding from raw document text and training data. Most neural IR methods focus on learning query-document relevance patterns, that is, knowledge about the search task. However, learning only relevance patterns requires large amounts of training data, and yet still doesn’t generalize well to tail queries or new search domains . These issues make pre-trained, generalpurpose text understanding models desirable. Pre-trained word representations such as word2vec

[REF3] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: Another line of research designs neural architectures to capture diverse matching features such as exact match signals and passage-level signals . How to understand the text content of the query/document is less explored. Most neural IR models represent text with word embeddings such as Word2Vec . arXiv:1905.09217v1 [cs.IR] 22 May 2019Figure 1: BERT sentence pair classification architecture .

[REF4] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: The models are pre-trained on a large number of documents so that the contextual representations encode general language patterns. Contextual neural language models have outperformed traditional word embeddings on a variety of NLP tasks [3, 8]. The deeper text understanding of contextual neural language models brings new possibilities to IR. This paper explores leveraging BERT (Bidirectional Encoder Representations from Transformers) for ad-hoc document retrieval. BERT is a state-of-the-art neural language model. It also fits well with search tasks.

[REF5] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: BERT-FirstP+Bing High High 0.333† 0.300† performance, confirming that text retrieval requires understanding both the text content and the search task. Simple domain adaptation of BERT leads to a pre-trained model with both types of knowledge that can improve related search tasks where labeled data are limited. 6 CONCLUSION Text understanding is a long-desired feature for text retrieval. Contextual neural language models open new possibilities for understanding word context and modeling language structures. This paper studies the effect of a recently-proposed deep neural language model, BERT, on ad-hoc document retrieval tasks. Adapting and fine-tuning BERT achieves high accuracy on two different search tasks, showing the effectiveness of BERT’s language modeling for IR.

[REF6] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: This paper studies the effect of a recently-proposed deep neural language model, BERT, on ad-hoc document retrieval tasks. Adapting and fine-tuning BERT achieves high accuracy on two different search tasks, showing the effectiveness of BERT’s language modeling for IR. The contextualized model brings large improvements to natural language queries. The corpus-trained language model can be complemented with search knowledge through simple domain adaptation, leading to a strong ranker that models both meanings of text and relevance in search. People have been trained to use keyword queries because bag-ofwords retrieval models cannot effectively extract key information from natural language. We found that queries written in natural language actually enable better search results when the system can model language structures. Our findings encourage additional research on search systems with natural language interfaces.

[REF7] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: It examines BERT models on two adhoc retrieval datasets with different characteristics. Experiments show that fine-tuning pre-trained BERT models with a limited amount of search data can achieve better performance than strong baselines. In contrast to observations from traditional retrieval models, longer natural language queries are able to outperform short keywords queries by large margins with BERT. Further analysis reveals that stopwords and punctuation, which are often ignored by traditional IR approaches, play a key role in understanding natural language queries by defining grammar structures and word dependencies. Finally, enhancing BERT with search knowledge from a large search log produces a pre-trained model equipped with knowledge about both text understanding and the search task, which benefits a related search task where labeled data are limited. 2 RELATED WORK Recent neural IR models have made promising progress in learning query-document relevance patterns. One line of research learns text presentations tailored for the search task

[REF8] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: Documents about foreign controllers or individuals are not relevant. may also be less effective on long text. We adopt a simple passagelevel approach for document retrieval. We split a document into overlapping passages. The neural ranker predicts the relevance of each passage independently. document score is the score of the first passage (BERT-FirstP), the best passage (BERT-MaxP), or the sum of all passage scores (BERT-SumP). For training, passage-level labels are not available in this work.

[REF9] - paperID: ./papers\_pdf/paper\_section/Interaction-focused\_Systems-Dealing\_with\_long\_texts/BIBREF45\_7a31e2dcbaa1cf6e9f76084793a02a2a4e4c2d15.pdf Title: Deeper Text Understanding for IR with Contextual Neural Language Modeling Chunk of text: Although intuitively, description queries should carry richer information, it is hard to fully utilize them in traditional bag-of-words methods due to difficulties in estimating term importance. Our results show that longer natural language queries are indeed more expressive than keywords, and the richer information can be effectively leveraged to improve search using a deep, contextualized neural language model. Further analysis of BERT’s ability to understand different types of search queries is given in Section 5.2. Robust04 vs. ClueWeb09-B. BERT models perform better on Robust04 than on ClueWeb09-B. This is probably due to that Robust04 is closer to the pre-trained model. Robust04 has well-written articles; its queries look for facts that depend largely on understanding text meaning. ClueWeb09-B documents are webpages that include tables, navigation bars, and other discontinuous text.

........................................................................................................................................................................................................

Title: Representation-focused Systems - Single Representations

Representation-focused systems in neural information retrieval (Neu-IR) aim to improve the effectiveness of retrieval by focusing on the representation learning of queries and documents. In this section, we discuss the use of single representations in representation-focused systems. Single representations refer to the encoding of queries and documents into low-dimensional embeddings, which capture their semantic meanings and enable efficient similarity search for ranking purposes.

Early research in Neu-IR suggested that interaction models, which handle term-level matches, are more effective but computationally expensive [REF0]. However, recent advancements have shown that single representation-based systems can achieve comparable effectiveness to interaction-based models. For instance, ANCE (Approximate Nearest Neighbor Negative Contrastive Learning) demonstrates that a properly trained representation-based BERT-Siamese model is as effective as an interaction-based BERT ranker [REF1]. This finding has motivated further research explorations in Neu-IR.

Deep learning techniques have been employed to enhance various components of sparse retrieval, such as term weighting, query expansion, and document expansion [REF0]. On the other hand, dense retrieval approaches take a different path by conducting retrieval purely in the embedding space using approximate nearest neighbor (ANN) search [REF0]. Dense retrieval systems have shown to achieve state-of-the-art accuracy and exhibit distinct behavior compared to classic retrieval methods [REF0].

Dense encodings in representation-focused systems offer several advantages over sparse representations. Dense encodings capture latent semantic information and can handle synonyms or paraphrases that consist of different tokens [REF3]. This capability enables better matching of queries and documents, even when they express similar meanings using different terms. Additionally, dense encodings are learnable through adjustment of embedding functions, allowing for task-specific representations [REF3]. Efficient retrieval using dense encodings can be achieved through specialized in-memory data structures and indexing schemes, such as maximum inner product search (MIPS) algorithms [REF3].

To evaluate the effectiveness of different input representations, existing model architectures are often utilized in representation-focused systems [REF4]. Relevance matching models, such as PACRR, KNRM, and DRMM, have shown effectiveness in neural relevance matching tasks without requiring extensive behavioral data [REF4]. These models serve as a benchmark to assess the performance of various input representations.

Precomputing Transformer Term Representations (PreTTR) is an approach that leverages transformer networks to generate term representations at index time [REF5]. During training, attention scores between queries and documents are masked to disallow interactions. At index time, the first layers of the transformer network are applied to each document, and the resulting term representations are stored. This approach enables end-to-end training and efficient processing at query time [REF5].

ANCE, a representation-focused system, constructs training negatives globally from the entire corpus to improve retrieval accuracy [REF1]. By using an asynchronously updated ANN index, ANCE eliminates the limitations of local negatives and enhances the convergence of dense retrieval models [REF1]. The improved retrieval accuracy of ANCE can benefit various language systems, including web search, OpenQA, and commercial search engines [REF1, REF6, REF7].

In summary, representation-focused systems that utilize single representations have shown promising results in improving retrieval effectiveness. These systems leverage deep learning techniques to encode queries and documents into low-dimensional embeddings, enabling efficient similarity search and better matching of semantic information. The advancements in representation-focused systems have opened up new avenues for research in Neu-IR.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF51\_c9b8593db099869fe7254aa1fa53f3c9073b0176.pdf Title: Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval Chunk of text: 7 RELATED WORK In early research on neural information retrieval (Neu-IR) (Mitra et al., 2018), a common belief was that the interaction models, those that specifically handle term level matches, are more effective though more expensive (Guo et al., 2016; Xiong et al., 2017; Nogueira & Cho, 2019). Many techniques are developed to reduce their cost, for example, distillation (Gao et al., 2020a) and caching (Humeau et al., 2020; Khattab & Zaharia, 2020; MacAvaney et al., 2020). ANCE shows that a properly trained representation-based BERT-Siamese is in fact as effective as the interaction-based BERT ranker. This finding will motivate many new research explorations in Neu-IR. Deep learning has been used to improve various components of sparse retrieval, for example, term weighting (Dai & Callan, 2019b), query expansion (Zheng et al., 2020), and document expansion (Nogueira et al., 2019). Dense Retrieval chooses a different path and conducts retrieval purely in the embedding space via ANN search (Lee et al., 2019; Chang et al., 2020; Karpukhin et al., 2020; Luan et al., 2020). This work demonstrates that a simple dense retrieval system can achieve SOTA accuracy, while also behaves dramatically different from classic retrieval.

[REF1] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF51\_c9b8593db099869fe7254aa1fa53f3c9073b0176.pdf Title: Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval Chunk of text: ANCE is orthogonal with those lines of research and focuses on the representation learning for dense retrieval. Its better retrieval accuracy can benefit many language systems. 8 CONCLUSION In this paper, we first provide theoretical analyses on the convergence of representation learning in dense retrieval. We show that under common conditions in text retrieval, the local negatives used in DR training are uninformative, yield low gradient norms, and contribute little to the learning convergence. We then propose ANCE to eliminate this bottleneck by constructing training negatives globally from the entire corpus. Our experiments demonstrate the advantage of ANCE in web search, OpenQA, and the production system of a commercial search engine. Our studies empirically validate our theory that ANCE negatives have much bigger gradient norms, reduce the stochastic gradient variance, and improve training convergence.

[REF2] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: [cs.IR] 19 Aug 2019a real-valued relevance estimate for the document to the query. Neural relevance ranking architectures generally use a similarity matrix as input S ∈ R |Q |× |D | , where each cell represents a similarity score between the query and document: Si,j = sim(qi ,dj). These similarity values are usually the cosine similarity score between the word vectors of each term in the query and document. 2.2 Contextualized similarity tensors Pretrained contextual language representations (such as those from ELMo and BERT ) are context sensitive; in contrast to more conventional pretrained word vectors (e.g., GloVe ) that generate a single word representation for each word in the vocabulary, these models generate a representation of each word based on its context in the sentence. For example, the contextualized representation of word bank would be different in bank deposit and river bank, while a pretrained word embedding model would always result in the same representation for this term. Given that these representations capture contextual information in the language, we investigate how these models can also benefit general neural ranking models.

[REF3] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: Conversely, the dense, latent semantic encoding is complementary to sparse representations by design. For example, synonyms or paraphrases that consist of completely different tokens may still be mapped to vectors close to each other. Consider the question “Who is the bad guy in lord of the rings?”, which can be answered from the context “Sala Baker is best known for portraying the villain Sauron in the Lord of the Rings trilogy.” A term-based system would have difficulty retrieving such a context, while a dense retrieval system would be able to better match “bad guy” with “villain” and fetch the correct context. Dense encodings are also learnable by adjusting the embedding functions, which provides additional flexibility to have a task-specific representation. With special in-memory data structures and indexing schemes, retrieval can be done efficiently using maximum inner product search (MIPS) algorithms (e.g., Shrivastava and Li (2014); Guo et al. (2016)).

[REF4] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF39\_1ec78c0ec945572673fabd50bf263870fe9d3601.pdf Title: CEDR: Contextualized Embeddings for Document Reranking Chunk of text: Models. Rather than building new models, in this work we use existing model architectures to test the effectiveness of various input representations. We evaluate our methods on three neural relevance matching methods: PACRR , KNRM , and DRMM . Relevance matching models have generally shown to be more effective than semantic matching models, while not requiring massive amounts of behavioral data (e.g., query logs).

[REF5] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF47\_0c3bdbad193ec8a5b1f4005dc1496e341a2025b4.pdf Title: Efficient Document Re-Ranking for Transformers by Precomputing Term Representations Chunk of text: These representations can then be stored and used at query time to finish the processing in conjunction with the query. This approach can be trained end-to-end by masking the attention across the query and document during training time (i.e., disallowing the document from attending to the query and vice versa.) We call this approach PreTTR (Precomputing Transformer Term Representations). A high-level overview of PreTTR is shown in Figure 1. At train time, a transformer network is fine-tuned for ad-hoc document ranking. This transformer network masks attention scores in the first l layers, disallowing interactions between the query and the document. At index time, each document in the collection is processed through the first l layers, and the resulting term representations are stored.

[REF6] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF51\_c9b8593db099869fe7254aa1fa53f3c9073b0176.pdf Title: Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval Chunk of text: Our experiments demonstrate the advantage of ANCE in three text retrieval scenarios: standard web search (Craswell et al., 2020), OpenQA (Rajpurkar et al., 2016; Kwiatkowski et al., 2019), and in a commercial search engine’s retrieval system. We also empirically validate our theory that the gradient norms on ANCE sampled negatives are much bigger than local negatives and thus improve the convergence of dense retrieval models. Our code and trained models are available at <https://aka.ms/ance>. 2 PRELIMINARIES In this section, we discuss the preliminaries of dense retrieval and its representation learning. Task Definition: Given a query q and a corpus C, the first stage retrieval is to find a set of documents relevant to the query D+ = {d1, ..., di , ..., dn} from C (|D+| |C|), which then serve as input to later more complex models (Croft et al., 2010). Instead of using sparse term matches and inverted index, Dense Retrieval calculates the retrieval score f() using similarities in a learned embedding space (Lee et al., 2019; Luan et al., 2020; Karpukhin et al., 2020): f(q, d) = sim(g(q; θ), g(d; θ)), (1) where g() is the representation model that encodes the query or document to dense embeddings. The encoder parameter θ provides the main capacity, often fine-tuned from pretrained transformers, e.g., BERT (Lee et al., 2019).

[REF7] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF51\_c9b8593db099869fe7254aa1fa53f3c9073b0176.pdf Title: Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval Chunk of text: This enlarged negative pool significantly improves unsupervised visual representation learning (Chen et al., 2020b). A parallel work (Xiong et al., 2020) improves DPR by sampling negatives from a memory bank (Wu et al., 2018) — in which the representations of negative candidates are frozen so more candidates can be stored. Instead of a bigger local pool, ANCE goes all the way along this trajectory and constructs negatives globally from the entire corpus, using an asynchronously updated ANN index. Besides being a real world application itself, dense retrieval is also a core component in many other language systems, for example, to retrieval relevant information for grounded language models (Khandelwal et al., 2019; Guu et al., 2020), extractive/generative QA (Karpukhin et al., 2020; Lewis et al., 2020b), and fact verification (Xiong et al., 2020), or to find paraphrase pairs for pretraining (Lewis et al., 2020a). There dense retrieval models are either frozen or optimized indirectly by signals from their end tasks. ANCE is orthogonal with those lines of research and focuses on the representation learning for dense retrieval. Its better retrieval accuracy can benefit many language systems.

[REF8] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF52\_7b577ba0e4230b2ac58d297b3d2cfc3d2f1aaace.pdf Title: Optimizing Dense Retrieval Model Training with Hard Negatives Chunk of text: To better deduce the users’ search intent and retrieve relevant items, the ranking algorithms are expected to conduct semantic matching between queries and documents , which is a challenging problem. In recent years, with the development of deep learning [6, 20, 28], especially representation learning techniques , many researchers have turned to the Dense Retrieval (DR) model to solve the semantic matching problem [10, 15, 18, 22]. In essence, DR attempts to encode queries and documents into low-dimension embeddings to better abstract their semantic meanings. With the learned embeddings, document index can be constructed and the query embedding can be adopted to perform efficient similarity search for online ranking. Previous studies showed that DR models achieve promising results on many IR-related tasks [7, 10, 15]. However, there are some unsolved but essential problems related to DR’s effectiveness and training efficiency1 . Firstly, though 1We focus on the training efficiency because the efficiency in the inference process is guaranteed by the maximum inner product search algorithms[14, 27]. arXiv:2104.08051v1

[REF9] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Single\_Representations/BIBREF47\_0c3bdbad193ec8a5b1f4005dc1496e341a2025b4.pdf Title: Efficient Document Re-Ranking for Transformers by Precomputing Term Representations Chunk of text: Furthermore, document token representations will need to have their context be fully captured in a way that is effective for the matching of the [CLS] representation. Interestingly, this setting blurs the line between representation-focused and interaction-focused neural models. Now we will consider the characteristics of each dataset. From Table 2, we find that the queries in the TREC WebTrack 2012 are typically shorter (mean: 2.0, median: 2, stdev: 0.8) than those from Robust (mean: 2.7, median: 3, stdev: 0.7). This results in queries that are more qualified, and may be more difficult to successfully represent in a single vector. To answer RQ4, we observe that the ranking effectiveness when combining with only a single transformer layer can vary depending on dataset characteristics. We find that in web collections (an environment where query-time latency is very important), it may be practical to use PreTTR in this way while maintaining high precision of the top-ranked documents.

........................................................................................................................................................................................................

Title: Representation-focused Systems - Multiple Representations

Representation-focused systems in neural information retrieval aim to effectively capture the semantic meaning and relevance between queries and documents. One approach to achieve this is by utilizing multiple representations, which can enhance the expressiveness and performance of the models [REF0].

One type of representation-focused system is the dual encoder model, which scores each document based on the inner product between its encoding and that of the query [REF1]. This model combines precise term overlap checking with semantic similarity computation, allowing it to generalize across related concepts. The advantage of the dual encoder model is its ability to efficiently handle large document collections using efficient algorithms for inner product search [REF2].

Another representation-focused system is the sparse-dense hybrid model, which combines the strengths of sparse retrieval models and learned dense representations [REF3]. Sparse retrieval models excel at precise term overlap checking, while learned dense representations are better at capturing semantic similarity. By leveraging both approaches, the hybrid model achieves improved performance in information retrieval tasks.

Cross-attentional architectures are another type of representation-focused system that can be viewed as a generalization of the multi-vector model [REF4]. These architectures allow for more expressive aggregation of vectors and can mimic sparse scoring functions. However, efficient retrieval at scale using fast nearest-neighbor search is challenging with cross-attention models [REF4].

In evaluations comparing different representation-focused systems, it has been found that full attentional architectures excel at reranking tasks but are not efficient enough for large-scale retrieval [REF5]. On the other hand, hybridized multi-vector encoders have shown promising results, outperforming state-of-the-art retrieval models in certain benchmarks [REF5].

To improve the performance of dense retrieval systems, further research has explored better training techniques and extensions to multi-vector representation systems [REF6]. These advancements have shown potential in enhancing the effectiveness of representation-focused models.

In summary, representation-focused systems that utilize multiple representations offer a way to capture both precise term overlap and semantic similarity in information retrieval tasks. Dual encoder models, sparse-dense hybrids, and cross-attentional architectures are among the approaches that have been explored in this area. These models aim to strike a balance between efficiency and effectiveness in capturing the relevance between queries and documents.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF55\_60b8ad6177230ad5402af409a6edb5af441baeb4.pdf Title: ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT Chunk of text: or kernels ). In the simplest case, they feed the neural network an interaction matrix that reects the similiarity between every pair of words across q and d. Further right, Figure 2 (c) illustrates a more powerful interaction-based paradigm, which models the interactions between words within as well as across q and d at the same time, as in BERT’s transformer architecture . ese increasingly expressive architectures are in tension. While interaction-based models (i.e., Figure 2 (b) and (c)) tend to be superior for IR tasks [8, 21], a representation-focused model—by isolating the computations among q and d—makes it possible to precompute document representations oine , greatly reducing the computational load per query. In this work, we observe that the ne-grained matching of interaction-based models and the precomputation of document representations of representation-based models can be combined by retaining yet judiciously delaying the query–document interaction.

[REF1] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: The dual encoder model scores each document by the inner product between its encoding and that of the query. Unlike full attentional architectures, which require extensive computation on each candidate document, the dual encoder can be easily applied to very large document collections thanks to efficient algorithms for inner product search; unlike untrained sparse retrieval models, it can exploit machine learning to generalize across related terms. To assess the relevance of a document to an information-seeking query, models must both (i) check for precise term overlap (for example, presence of key entities in the query) and (ii) compute semantic similarity generalizing across rearXiv:2005.00181v3 [cs.CL] 16 Feb 2021lated concepts. Sparse retrieval models excel at the first sub-problem, while learned dual encoders can be better at the second. Recent history in NLP might suggest that learned dense representations should always outperform sparse features overall, but this is not necessarily true: as shown in Figure 1, the BM25 model (Robertson et al., 2009) can outperform a dual encoder based on BERT, particularly on longer documents and on a task that requires precise detection of word overlap.1

[REF2] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: Craswell et al., 2020). However, this pipeline suffers from a strict upper bound imposed by any recall errors in the first-stage retrieval model: for example, the recall@1000 for BM25 reported by Yan et al. (2020) is 69.4. A promising alternative is to perform first-stage retrieval using learned dense low-dimensional encodings of documents and queries (Huang et al., 2013; Reimers and Gurevych, 2019; Gillick et al., 2019; Karpukhin et al., 2020). The dual encoder model scores each document by the inner product between its encoding and that of the query. Unlike full attentional architectures, which require extensive computation on each candidate document, the dual encoder can be easily applied to very large document collections thanks to efficient algorithms for inner product search; unlike untrained sparse retrieval models, it can exploit machine learning to generalize across related terms.

[REF3] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF55\_60b8ad6177230ad5402af409a6edb5af441baeb4.pdf Title: ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT Chunk of text: introduced SNRM, a representationfocused IR model that encodes each query and each document as a single, sparse high-dimensional vector of “latent terms”. By producing a sparse-vector representation for each document, SNRM is able to use a traditional IR inverted index for representing documents, allowing fast end-to-end retrieval. Despite highly promising results and insights, SNRM’s eectiveness is substantially outperformed by the state of the art on the datasets with which it was evaluated (e.g., see [18, 38]). While SNRM employs sparsity to allow using inverted indexes, we relax this assumption and compare a (dense) BERT-based representation-focused model against our late-interaction ColBERT in our ablation experiments in §4.4. For a detailed overview of existing neural ranking models, we refer the readers to two recent surveys of the literature [8, 21]. Language Model Pretraining for IR. Recent work in NLU emphasizes the importance pre-training language representation models in an unsupervised fashion before subsequently ne-tuning them on downstream tasks.

[REF4] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: Cross-attentional architectures can be viewed as a generalization of the multivector model: (1) set m = Tmax (one vector per token); (2) compute one vector per token in the query; (3) allow more expressive aggregation 3Here we use (d, q) rather than (x, y) because we describe vector encodings rather than token sequences.over vectors than the simple max employed above. Any sparse scoring function (e.g., BM25) can be mimicked by a cross-attention model, which need only compute identity between individual words; this can be achieved by random projection word embeddings whose dimension is proportional to the log of the vocabulary size. By definition, the required representation also grows linearly with the number of tokens in the passage and query. As with the POLY-ENCODER, retrieval in the crossattention model cannot be performed efficiently at scale using fast nearest-neighbor search. In contemporaneous work, Khattab and Zaharia (2020) propose an approach with TY vectors per query and TX vectors per document, using a simple sum-of-max for aggregation of the inner products. They apply this approach to retrieval via reranking results of TY nearest-neighbor searches.

[REF5] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: We compare the performance of dual encoders, multi-vector encoders, and their sparse-dense hybrids with classical sparse retrieval models and attentional neural networks, as well as state-ofthe-art published results where available. Our evaluations include open retrieval benchmarks (MS MARCO passage and document), and passage retrieval for question answering (Natural Questions). We confirm prior findings that full attentional architectures excel at reranking 1 See § 4 for experimental details. tasks, but are not efficient enough for large-scale retrieval. Of the more efficient alternatives, the hybridized multi-vector encoder is at or near the top in every evaluation, outperforming stateof-the-art retrieval results in MS MARCO. Our code is publicly available at [https://github](https://github/). com/google-research/language/tree/ master/language/multivec. 2 Analyzing dual encoder fidelity A query or a document is a sequence of words drawn from some vocabulary V. Throughout this section we assume a representation of queries and documents typically used in sparse bag-of-words models: each query q and document d is a vector in R v where v is the vocabulary size.

[REF6] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF56\_2d7a784a093615d00d4ac0a7b5763a15d86d4996.pdf Title: COIL: Revisit Exact Lexical Match in Information Retrieval with Contextualized Inverted List Chunk of text: Later researches show that dense retrieval systems can be further improved by better training (Xiong et al., 2020; Gao et al., 2021b). Single vector systems have also been extended to multi-vector representation systems. Polyencoder (Humeau et al., 2020) encodes queries into a set of vectors. Similarly, Me-BERT (Luan et al., 2020) represents documents with a set of vectors. A concurrent work ColBERT (Figure 1c) use multiple vectors to encode both queries and documents (Khattab and Zaharia, 2020). In particular, it represents a documents with all its terms’ vectors and a query with an expanded set of term vectors. It then computes all-to-all (Cartesian) soft match between the tokens.

[REF7] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: = 1: the representations for queries and documents are the top layer representations at the [CLS] token. This approach is widely used for retrieval (Lee et al., 2019; Reimers and Gurevych, 2019; Humeau et al., 2020; Xiong et al., 2020).4 For lower-dimensional encodings, we learn downprojections from d = 768 to k ∈ 32, 64, 128, 512, 5 4Based on preliminary experiments with pooling strategies we use the [CLS] vectors (without the feed-forward projection learned on the next sentence prediction task). 5We experimented with adding a similar layer for d = 768, but this did not offer empirical gains.implemented as a single feed-forward layer, followed by layer normalization. All parameters are fine-tuned for the retrieval tasks. We refer to these models as DE-BERT-k. Cross-Attentional BERT.

[REF8] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: We create a dataset with one million queries and evaluate retrieval against four document collections Dl , for l ∈ 50, 100, 200, 400. Each Dl contains three million documents of maximum length l tokens. In addition to original Wikipedia passages, each Dl contains synthetic distractor documents, which contain the large majority of words in x but differ by one or two tokens. 5K queries are used for evaluation, leaving the rest for training and validation. Although checking containment is a straightforward machine learning task, it is a good testbed for assessing the fidelity of compressive neural models. BM25-bi achieves over 95 MRR@10 across collections for this task. Figure 4 (left) shows test set results on reranking, where models need to select one of 200 passages (top 100 BM25-bi and 100 random candi-50 100 200 400 40 60 80 100 MRR@10 Passage Ranking for ICT 50 100 200 400 40 60 80 100 MRR@10 Passage Retrieval for ICT Cross-Attention DE-BERT-32 DE-BERT-64 DE-BERT-128 DE-BERT-512 DE-BERT-768 ME-BERT-64 ME-BERT-768 HYBRID-ME-BERT-uni HYBRID-ME-BERT-bi BM25-uni BM25-bi Figure 4: Results on the containing passage ICT task as maximum passage length varies (50 to 400 tokens).

[REF9] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Multiple\_Representations/BIBREF54\_050050e30d0f162c4dd87c1aac8d37df266e4c93.pdf Title: Sparse, Dense, and Attentional Representations for Text Retrieval Chunk of text: T. The relevance score is defined as maxj=1...mhf (1)(x), f(m) j (y)i. Although this scoring function is not a dual encoder, the search for the highest-scoring document can be implemented efficiently with standard approximate nearest-neighbor search by adding multiple (m) entries for each document to the search index data structure. If some vector f (m) j (y) yields the largest inner product with the query vector f (1)(x), it is easy to show the corresponding document must be the one that maximizes the relevance score ψ (m) (x, y). The size of the index must grow by a factor of m, but due to the efficiency of contemporary approximate nearest neighbor and maximum inner product search, the time complexity can be sublinear in the size of the index (Andoni et al., 2019; Guo et al., 2016b). Thus, a model using m vectors of size k to represent documents is more efficient at run-time than a dual encoder that uses a single vector of size mk. This efficiency is a key difference from the POLY-ENCODER (Humeau et al., 2020), which computes a fixed number of vectors per query, and aggregates them by softmax attention against document vectors. Yang et al.

........................................................................................................................................................................................................

Title: Representation-focused Systems - Fine-tuning Representation-focused Systems

Representation-focused systems in neural information retrieval aim to improve the retrieval performance by leveraging dense representations of text. These systems complement sparse vector space models by capturing semantic relationships between words and phrases [REF0]. Dense encodings allow for better matching of synonyms and paraphrases, even if they consist of different tokens, by mapping them to vectors close to each other. For example, a dense retrieval system can effectively match the query "Who is the bad guy in Lord of the Rings?" with the context "Sala Baker is best known for portraying the villain Sauron in the Lord of the Rings trilogy" [REF0].

Fine-tuning representation-focused systems involves adjusting the embedding functions to learn task-specific representations [REF0]. This flexibility allows for the optimization of the dense encodings to better suit the retrieval task at hand. Efficient retrieval using dense representations can be achieved through the use of in-memory data structures and indexing schemes, such as maximum inner product search (MIPS) algorithms [REF0].

While dense representations alone may be inferior to sparse ones, they have shown effectiveness in passage or dialogue re-ranking tasks [REF2]. Pretrained models and cross-attention mechanisms have been utilized to enhance the performance of dense encodings in these tasks [REF2]. Additionally, alternative approaches have been proposed, such as late-interaction operators and direct retrieval of answer phrases, to improve the efficiency and effectiveness of dense retrieval for open-domain question answering [REF2].

Dense Passage Retrieval (DPR) has emerged as an important module in recent work, demonstrating its potential for improving retrieval performance [REF3]. By leveraging hard negatives and constructing different sets of examples in each training iteration, DPR has shown further improvements in retrieval performance [REF3]. Moreover, the combination of DPR with generation models, such as BART and T5, has achieved good performance in open-domain question answering and other knowledge-intensive tasks [REF3].

In the training stage of representation-focused systems, models are optimized using mini-batch based stochastic gradient descent (SGD) [REF4]. The convergence of the training process is typically observed within a few epochs over the entire training data [REF4]. Experimental evaluations of these systems have been conducted on large-scale real-world datasets, showcasing their performance in web document ranking tasks [REF4].

The introduction of deep learning methods has revolutionized semantic modeling in information retrieval [REF5]. Latent semantic models, such as LSA, have been widely used for query-document matching [REF5]. These models project documents and queries into low-dimensional concept vectors, enabling similarity calculations based on cosine similarity scores [REF5]. Translation models trained on clicked query-document pairs have also provided an alternative approach to semantic matching [REF5].

Sparse vector space models, such as TF-IDF and BM25, have been the standard methods applied to various question answering tasks [REF6]. However, the use of dense vector representations has gained popularity due to their ability to assign high similarity scores to semantically relevant text pairs, even without exact token matching [REF6]. Dense representations, when combined with cross-attention mechanisms, have shown effectiveness in passage or dialogue re-ranking tasks [REF6].

In the context of question answering systems, representation-focused systems consist of a retriever and a neural reader [REF7]. The retriever retrieves relevant passages based on the query, and the reader assigns passage selection scores and extracts answer spans from the retrieved passages [REF7]. Cross-attention between the question and the passage is utilized to rerank the passages and select the most relevant information [REF7].

Building dense embeddings on a large number of passages can be resource-intensive but can be parallelized for efficiency [REF8]. However, constructing indexes for these embeddings can be time-consuming, and alternative methods such as inverted indexes using Lucene have been explored as more cost-effective options [REF8].

In summary, fine-tuning representation-focused systems in neural information retrieval involves adjusting embedding functions to optimize dense encodings for specific retrieval tasks. These systems complement sparse vector space models and have shown effectiveness in improving retrieval performance. The combination of dense retrieval with other techniques, such as cross-attention mechanisms and translation models, further enhances the capabilities of representation-focused systems in question answering tasks.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: Conversely, the dense, latent semantic encoding is complementary to sparse representations by design. For example, synonyms or paraphrases that consist of completely different tokens may still be mapped to vectors close to each other. Consider the question “Who is the bad guy in lord of the rings?”, which can be answered from the context “Sala Baker is best known for portraying the villain Sauron in the Lord of the Rings trilogy.” A term-based system would have difficulty retrieving such a context, while a dense retrieval system would be able to better match “bad guy” with “villain” and fetch the correct context. Dense encodings are also learnable by adjusting the embedding functions, which provides additional flexibility to have a task-specific representation. With special in-memory data structures and indexing schemes, retrieval can be done efficiently using maximum inner product search (MIPS) algorithms (e.g., Shrivastava and Li (2014); Guo et al. (2016)).

[REF1] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: While a simple dual-encoder approach can be made to work surprisingly well, we showed that there are some critical ingredients to training a dense retriever successfully. Moreover, our empirical analysis and ablation studies indicate that more complex model frameworks or similarity functions do not necessarily provide additional values. As a result of improved retrieval performance, we obtained new state-of-the-art results on multiple open-domain question answering benchmarks. Acknowledgments We thank the anonymous reviewers for their helpful comments and suggestions.

[REF2] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: The dense representation alone, however, is typically inferior to the sparse one. While not the focus of this work, dense representations from pretrained models, along with cross-attention mechanisms, have also been shown effective in passage or dialogue re-ranking tasks (Nogueira and Cho, 2019; Humeau et al., 2020). Finally, a concurrent work (Khattab and Zaharia, 2020) demonstrates the feasibility of full dense retrieval in IR tasks. Instead of employing the dual-encoder framework, they introduced a late-interaction operator on top of the BERT encoders. Dense retrieval for open-domain QA has been explored by Das et al. (2019), who propose to retrieve relevant passages iteratively using reformulated question vectors. As an alternative approach that skips passage retrieval, Seo et al. (2019) propose to encode candidate answer phrases as vectors and directly retrieve the answers to the input questions efficiently. Using additional pretraining with the objective that matches surrogates of questions and relevant passages, Lee et al. (2019) jointly train the question encoder and reader.

[REF3] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: DPR has also been used as an important module in very recent work. For instance, extending the idea of leveraging hard negatives, Xiong et al. (2020a) use the retrieval model trained in the previous iteration to discover new negatives and construct a different set of examples in each training iteration. Starting from our trained DPR model, they show that the retrieval performance can be further improved. Recent work (Izacard and Grave, 2020; Lewis et al., 2020b) have also shown that DPR can be combined with generation models such as BART (Lewis et al., 2020a) and T5 (Raffel et al., 2019), achieving good performance on open-domain QA and other knowledge-intensive tasks. 8 Conclusion In this work, we demonstrated that dense retrieval can outperform and potentially replace the traditional sparse retrieval component in open-domain question answering. While a simple dual-encoder approach can be made to work surprisingly well, we showed that there are some critical ingredients to training a dense retriever successfully. Moreover, our empirical analysis and ablation studies indicate that more complex model frameworks or similarity functions do not necessarily provide additional values.

[REF4] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF58\_fdb813d8b927bdd21ae1858cafa6c34b66a36268.pdf Title: Learning Deep Structured Semantic Models for Web Search using Clickthrough Data Chunk of text: In the training stage, we optimize the model using mini-batch based stochastic gradient descent (SGD). Each mini-batch consists of 1024 training samples. We observed that the DNN training usually converges within 20 epochs (passes) over the entire training data. 4. EXPERIMENTS We evaluated the DSSM, proposed in Section 3, on the Web document ranking task using a real-world data set. In this section, we first describe the data set on which the models are evaluated. Then, we compare the performances of our best model against other state of the art ranking models. We also investigate the break-down impact of the techniques proposed in Section 3. 4.1 Data Sets and Evaluation Methodology We have evaluated the retrieval models on a large-scale real world data set, called the evaluation data set henceforth.

[REF5] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF58\_fdb813d8b927bdd21ae1858cafa6c34b66a36268.pdf Title: Learning Deep Structured Semantic Models for Web Search using Clickthrough Data Chunk of text: The second is the introduction of deep learning methods for semantic modeling . 2.1 Latent Semantic Models and the Use of Clickthrough Data The use of latent semantic models for query-document matching is a long-standing research topic in the IR community. Popular models can be grouped into two categories, linear projection models and generative topic models, which we will review in turn. The most well-known linear projection model for IR is LSA . By using the singular value decomposition (SVD) of a document-term matrix, a document (or a query) can be mapped to a low-dimensional concept vector ̂ , where the is the projection matrix. In document search, the relevance score between a query and a document, represented respectively by term vectors and , is assumed to be proportional to their cosine similarity score of the corresponding concept vectors ̂ and ̂ , according to the projection matrix ̂ ̂ ̂ ̂ (1) In addition to latent semantic models, the translation models trained on clicked query-document pairs provide an alternative approach to semantic matching . Unlike latent semantic models, the translation-based approach learns translation relationships directly between a term in a document and a term in a query.

[REF6] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: Strong sparse vector space models like TF-IDF or BM25 havebeen used as the standard method applied broadly to various QA tasks (e.g., Chen et al., 2017; Yang et al., 2019a,b; Nie et al., 2019; Min et al., 2019a; Wolfson et al., 2020). Augmenting text-based retrieval with external structured information, such as knowledge graph and Wikipedia hyperlinks, has also been explored recently (Min et al., 2019b; Asai et al., 2020). The use of dense vector representations for retrieval has a long history since Latent Semantic Analysis (Deerwester et al., 1990). Using labeled pairs of queries and documents, discriminatively trained dense encoders have become popular recently (Yih et al., 2011; Huang et al., 2013; Gillick et al., 2019), with applications to cross-lingual document retrieval, ad relevance prediction, Web search and entity retrieval. Such approaches complement the sparse vector methods as they can potentially give high similarity scores to semantically relevant text pairs, even without exact token matching. The dense representation alone, however, is typically inferior to the sparse one. While not the focus of this work, dense representations from pretrained models, along with cross-attention mechanisms, have also been shown effective in passage or dialogue re-ranking tasks (Nogueira and Cho, 2019; Humeau et al., 2020).

[REF7] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: Besides the retriever, our QA system consists of a neural reader that outputs the answer to the question. Given the top k retrieved passages (up to 100 in our experiments), the reader assigns a passage selection score to each passage. In addition, it extracts an answer span from each passage and assigns a span score. The best span from the passage with the highest passage selection 10FAISS configuration: we used HNSW index type on CPU, neighbors to store per node = 512, construction time search depth = 200, search depth = 128. score is chosen as the final answer. The passage selection model serves as a reranker through crossattention between the question and the passage. Although cross-attention is not feasible for retrieving relevant passages in a large corpus due to its nondecomposable nature, it has more capacity than the dual-encoder model sim(q, p) as in Eq.

[REF8] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF50\_79cd9f77e5258f62c0e15d11534aea6393ef73fe.pdf Title: Dense Passage Retrieval for Open-Domain Question Answering Chunk of text: Computing dense embeddings on 21-million passages is resource intensive, but can be easily parallelized, taking roughly 8.8 hours on 8 GPUs. However, building the FAISS index on 21-million vectors on a single server takes 8.5 hours. In comparison, building an inverted index using Lucene is much cheaper and takes only about 30 minutes in total. 6 Experiments: Question Answering In this section, we experiment with how different passage retrievers affect the final QA accuracy. 6.1 End-to-end QA System We implement an end-to-end question answering system in which we can plug different retriever systems directly. Besides the retriever, our QA system consists of a neural reader that outputs the answer to the question. Given the top k retrieved passages (up to 100 in our experiments), the reader assigns a passage selection score to each passage.

[REF9] - paperID: ./papers\_pdf/paper\_section/Representation-focused\_Systems-Fine-tuning\_Representation-focused\_Systems/BIBREF58\_fdb813d8b927bdd21ae1858cafa6c34b66a36268.pdf Title: Learning Deep Structured Semantic Models for Web Search using Clickthrough Data Chunk of text: On the other hand, the training of DPM involves large-scale matrix multiplications. The sizes of these matrices often grow quickly with the vocabulary size, which could be of an order of millions in Web search tasks. In order to make the training time tolerable, the vocabulary was pruned aggressively. Although a small vocabulary makes the models trainable, it leads to suboptimal performance. In the second line of research, Salakhutdinov and Hinton extended the semantic modeling using deep auto-encoders . Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

........................................................................................................................................................................................................

Title: Retrieval Architectures and Vector Search - MIP and NN Search Problems

In the field of Neural Information Retrieval, one important problem is Maximum Inner Product Search (MIPS) [REF0]. MIPS involves finding a data vector from a collection of "database" vectors that maximizes the inner product with a given query vector [REF0]. This problem arises in various applications such as matrix-factorization based recommendation systems, multi-class prediction, structural SVM problems, and vision problems [REF0]. To efficiently find approximate solutions for MIPS, Shrivastava and Li (2014a) propose constructing a Locality Sensitive Hash (LSH) for inner product similarity [REF0].

LSH is a popular tool for approximate nearest neighbor search and has been widely used in various settings [REF0]. However, it is important to note that no LSH, whether symmetric or asymmetric, is possible for inner product similarity over the entire space R^d [REF1]. Nevertheless, in the context of MIPS, certain assumptions can be made without loss of generality [REF1]. First, the query vector can be normalized, as the norm of the query does not affect the argmax in MIPS [REF1]. Second, the database vectors can be bounded inside the unit sphere, as rescaling all vectors without changing the argmax is possible [REF1].

While symmetric LSH is not possible for inner product similarity over the entire space R^d, Shrivastava and Li (2014a) propose two distinct mappings that yield an asymmetric LSH for MIPS [REF3]. However, it is worth noting that their asymmetric LSH is only valid when queries are normalized and data vectors are bounded [REF3]. In contrast, a symmetric LSH is possible for the MIPS setting with normalized queries and bounded database vectors [REF3]. Furthermore, a universal symmetric LSH is achievable under these assumptions [REF3]. It is important to consider the need for an asymmetric view of MIPS, even though an asymmetric hash is not required in the MIPS setting [REF6].

The comparison between LSH-based methods and tree-based methods, such as cone trees, is an ongoing topic of research [REF5]. While the exact regimes where LSH-based methods are superior to tree-based methods are not fully established, the goal is to understand which LSH to use and why in situations where tree-based methods are not practical [REF5]. Shrivastava and Li (2014a) argue that tree-based methods are impractical in high dimensions, while the performance of LSH-based methods is relatively independent of the dimensionality of the data [REF5].

In conclusion, the problem of Maximum Inner Product Search (MIPS) in Neural Information Retrieval involves finding a data vector that maximizes the inner product with a given query vector. While LSH-based methods, including symmetric and asymmetric LSH, are not possible for inner product similarity over the entire space R^d, they can be applied in the context of MIPS with certain assumptions. The comparison between LSH-based methods and tree-based methods is an ongoing area of research, aiming to understand the practicality and effectiveness of different LSH approaches in various scenarios.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: Introduction Following Shrivastava and Li (2014a), we consider the problem of Maximum Inner Product Search (MIPS): given a collection of “database” vectors S ⊂ R d and a query q ∈ R d , find a data vector maximizing the inner product with the query: p = arg max x∈S q ⊤x (1) MIPS problems of the form (1) arise, e.g. when using matrix-factorization based recommendation systems (Koren et al., 2009; Srebro et al., 2005; Cremonesi et al., 2010), in multi-class prediction (Dean et al., 2013; Jain et al., 2009) and structural SVM (Joachims, 2006; Joachims et al., 2009) problems and in vision problems when scoring filters based on their activations (Dean et al., 2013) (see Shrivastava and Li, 2014a, for more about MIPS). In order to efficiently find approximate MIPS solutions, Shrivastava and Li (2014a) suggest constructing a Locality Sensitive Hash (LSH) for inner product “similarity”. Proceedings of the 31 st International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s). Locality Sensitive Hashing (Indyk and Motwani, 1998) is a popular tool for approximate nearest neighbor search and is also widely used in other settings (Gionis et al., 1999; Datar et al., 2004; Charikar, 2002).

[REF1] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: For completeness, we also include in AppendixB a full definition of the max-norm and margin complexity, as well as the bounds on the max-norm and margin complexity used in the proof above. 4. Maximum Inner Product Search We saw that no LSH, nor ALSH, is possible for inner product similarity over the entire space R d . Fortunately, this is not required for MIPS. As pointed out by Shrivastava and Li (2014a), we can assume the following without loss of generality: • The query q is normalized: Since given a vector q, the norm kqk does not affect the argmax in (1), we can assume kqk = 1 always. • The database vectors are bounded inside the unit sphere: We assume kxk ≤ 1 for all x ∈ S. Otherwise we can rescale all vectors without changing the argmax. We cannot, of course, assume the vectors x are normalized.

[REF2] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: = 1 − cos−1 (q ⊤x) π . As in the proof of Theorem 4.2, monotonicity of 1 − cos−1 (x) π establishes the desired ALSH properties. Shrivastava and Li (2015) also showed how a modification of SIMPLE-ALSH can be used for searching similarity measures such as set containment and weighted Jaccard similarity. 6. Conclusion We provide a complete characterization of when symmetric and asymmetric LSH are possible for inner product similarity: • Over R d , no symmetric nor asymmetric LSH is possible. • For the MIPS setting, with normalized queries kqk = 1 and bounded database vectors kxk ≤ 1, a universal symmetric LSH is possible. • When queries and database vectors are bounded but not normalized, a symmetric LSH is not possible, but a universal asymmetric LSH is.

[REF3] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: Although the exact regimes under which LSH-based methods are superior to tree-based methods and vice versa are not fully established yet, the goal of this paper is to analyze different LSH methods and compare them with each other, rather than comparing to tree-based methods, so as to understand which LSH to use and why, in those regimes where tree-based methods are not practical. Considering MIPS, Shrivastava and Li (2014a) argue that there is no symmetric LSH for inner product similarity, and propose two distinct mappings, one of database objects and the other for queries, which yields an asymmetric LSH for MIPS. But the caveat is that they consider different spaces in their positive and negative results: they show nonexistence of a symmetric LSH over the entire space R d , but their asymmetric LSH is only valid when queries are normalized and data vectors are bounded. Thus, they do notOn Symmetric and Asymmetric LSHs for Inner Product Search actually show a situation where an asymmetric hash succeeds where a symmetric hash is not possible. In fact, in Section 4 we show a simple symmetric LSH that is also valid under the same assumptions, and it even enjoys improved theoretical guarantees and empirical performance! This suggests that asymmetry might actually not be required nor helpful for MIPS. Motivated by understanding the power of asymmetry, and using this understanding to obtain the simplest and best possible LSH for MIPS, we conduct a more careful study of LSH for inner product similarity.

[REF4] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: This suggests that asymmetry might actually not be required nor helpful for MIPS. Motivated by understanding the power of asymmetry, and using this understanding to obtain the simplest and best possible LSH for MIPS, we conduct a more careful study of LSH for inner product similarity. A crucial issue here is what is the space of vectors over which we would like our LSH to be valid. First, we show that over the entire space R d , not only is there no symmetric LSH, but there is also no asymmetric LSH either (Section 3). Second, as mentioned above, when queries are normalized and data is bounded, a symmetric LSH is possible and there is no need for asymmetry. But when queries and data vectors are bounded and queries are not normalized, we do observe the power of asymmetry: here, a symmetric LSH is not possible, but an asymmetric LSH exists (Section 5). As mentioned above, our study also yields an LSH for MIPS, which we refer to as SIMPLE-LSH, which is not only symmetric but also parameter-free and enjoys significantly better theoretical and empirical compared to L2-ALSH(SL) proposed by Shrivastava and Li (2014a).

[REF5] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: ≈ P[h(x) = g(y)] (Neyshabur et al., 2013; 2014). Neyshabur et al. showed that even when the similarity sim(x, y) is entirely symmetric, asymmetry in the hash may enable obtaining an LSH when a symmetric LSH is not possible, or enable obtaining a much better LSH yielding shorter and more accurate hashes. Several tree-based methods have also been proposed for inner product search (Ram and Gray, 2012; Koenigstein et al., 2012; Curtin et al., 2013). Shrivastava and Li (2014a) argue that tree-based methods, such as cone trees, are impractical in high dimensions while the performance of LSH-based methods is in a way independent of dimension of the data. Although the exact regimes under which LSH-based methods are superior to tree-based methods and vice versa are not fully established yet, the goal of this paper is to analyze different LSH methods and compare them with each other, rather than comparing to tree-based methods, so as to understand which LSH to use and why, in those regimes where tree-based methods are not practical. Considering MIPS, Shrivastava and Li (2014a) argue that there is no symmetric LSH for inner product similarity, and propose two distinct mappings, one of database objects and the other for queries, which yields an asymmetric LSH for MIPS.

[REF6] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: Furthermore, even in the MIPS setting when queries are normalized (the second setting), the asymmetric hashes suggested by Shrivastava and Li (2014a;b) are not universal and require tuning parameters specific to S, c, in contrast to SIMPLE-LSH which is symmetric, parameter-free and universal. It is important to emphasize that even though in the MIPS setting an asymmetric hash, as we define here, is not needed, an asymmetric view of the problem is required. In particular, to use a symmetric hash, one must normalize the queries but not the database vectors, which can legitimately be viewed as an asymmetric operation which is part of the����On Symmetric and Asymmetric LSHs for Inner Product Search hash (though then the hash would not be, strictly speaking, an ALSH). In this regard Shrivastava and Li (2014a) do indeed successfully identify the need for an asymmetric view of MIPS, and provide the first practical ALSH for the problem. ACKNOWLEDGMENTS This research was partially funded by NSF award IIS1302662.

[REF7] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: They furthermore calculate the hashing quality ρ as a function of m, U and r, and numerically find the optimal ρ over a grid of possible values for m, U and r, for each choice of S, c. Before moving on to presenting a symmetric hash for the problem, we note that L2-ALSH(SL) is not universal (as defined at the end of Section 2). That is, not only might the optimal m, U and r depend on S, c, but in fact there is no choice of the parameters m and U that yields an ALSH for all S, c, or even for all ratios c for some specific threshold S or for all thresholds S for some specific ratio c. This is unfortunate, since in MIPS problems, the relevant threshold S is the maximal inner product maxx∈S q ⊤x (or the threshold inner product if we are interested in the “top-k” hits), which typically varies with the query. It is therefore desirable to have a single hash that works for all thresholds. Lemma 1. For any m, U, r, and for any 0 < S < 1 and 1 − U 2m+1−1 (1 − S 2m+1 ) 2S ≤ c < 1, L2-ALSH(SL) is not an (S, cS)-ALSH for inner product similarity over X• = {x|kxk ≤ 1} and Y◦ = {q|kqk = 1}.

[REF8] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: That is, for each user i we would like to find the top T movies, i.e. the T movies with highest hLi , Rj i, for different values of T . To do so, for each hash family, we generate hash codes of length K, for varying lengths K, for all movies and a random selection of 60000 users (queries). For each user, we sort movies in ascending order of hamming distance between the user hash and movie hash, breaking up ties randomly. For each of several values of T and K we calculate precision-recall curves for recalling the top T movies, averaging the precision-recall values over the 60000 randomly selected users. In Figures 2 and 3, we plot precision-recall curves of retrieving top T items by hash code of length K for Netflix and Movielens datasets where T ∈ {1, 5, 10} and K ∈ {64, 128, 256, 512}. For L2-ALSH(SL)

[REF9] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-MIP\_and\_NN\_Search\_Problems/BIBREF62\_5b0a88bdec473552c6a386cd94fdac53c74b79a8.pdf Title: On Symmetric and Asymmetric LSHs for Inner Product Search Chunk of text: In Figures 2 and 3, we plot precision-recall curves of retrieving top T items by hash code of length K for Netflix and Movielens datasets where T ∈ {1, 5, 10} and K ∈ {64, 128, 256, 512}. For L2-ALSH(SL) we used m = 3, U = 0.83, r = 2.5, suggested by the authors and used in their empirical evaluation. For SIGN-ALSH(SL) we used two different settings of the parameters suggested by Shrivastava and Li (2014b): m = 2, U = 0.75 and m = 3, U = 0.85. SIMPLE-LSH does not require any parameters. As can be seen in the Figures, SIMPLE-LSH shows a dramatic empirical improvement over L2-ALSH(SL).

........................................................................................................................................................................................................

Title: Retrieval Architectures and Vector Search - Locality sensitive hashing approaches

Locality sensitive hashing (LSH) is a well-known indexing method for Approximate Nearest-Neighbor (ANN) search, which aims to find objects that have similar characteristics to a given query object [REF0]. LSH uses hash functions to map similar objects into the same hash buckets with high probability [REF5]. The basic LSH indexing method involves checking the buckets to which the query object is hashed, which typically requires a large number of hash tables (hundreds) to achieve good search quality [REF0]. However, this approach can be computationally expensive and may not be efficient for high-dimensional datasets [REF3].

To address these limitations, Bawa et al. proposed the LSH Forest indexing method, which represents each hash table by a prefix tree [REF0]. This allows for the adaptation of the number of hash functions per table based on different approximation distances [REF0]. By using vector approximations or bounding rectangle approximations, LSH Forest can effectively prune the search space and improve search efficiency [REF1].

LSH has been widely studied in the context of high-dimensional similarity search [REF3]. It has been shown that traditional indexing methods based on space partitioning, such as R-tree, K-D tree, SR-tree, navigating-nets, and cover-tree, are slower than the brute-force linear-scan approach when the dimensionality exceeds a certain threshold [REF4]. In contrast, LSH has emerged as the best-known indexing method for high-dimensional similarity search [REF3].

LSH functions have the property that objects close to each other have a higher probability of colliding in the hash buckets [REF5]. This property allows LSH to efficiently select candidate objects for a given query by hashing the query point and retrieving elements stored in the corresponding buckets [REF6]. The search quality of LSH is measured by comparing the distances of the retrieved objects to the query with the distances of the true nearest neighbors [REF2].

Recent advancements in LSH include the multi-probe LSH method, which improves both space and time efficiency compared to the basic and entropy-based LSH methods [REF7]. The multi-probe LSH method reduces the number of hash tables required while achieving similar search quality and query time efficiencies [REF7]. By computing a non-overlapped bucket sequence based on the probability of containing similar objects, the multi-probe LSH method avoids the redundancy of hashed buckets by perturbed queries [REF8].

In conclusion, locality sensitive hashing (LSH) is a powerful indexing method for Approximate Nearest-Neighbor (ANN) search in high-dimensional spaces. It efficiently maps similar objects into the same hash buckets and allows for effective pruning of the search space. Recent advancements, such as the multi-probe LSH method, have further improved the space and time efficiency of LSH for similarity search tasks [REF7].

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: Please see [7, 15, 12] for some survey. Here, we focus on locality sensitive hashing techniques that are most relevant to our work. Locality sensitive hashing (LSH), introduced by Indyk and Motwani, is the best-known indexing method for ANN search. Theoretical lower bounds for LSH have also been studied [21, 1]. The basic LSH indexing method only checks the buckets to which the query object is hashed and usually requires a large number of hash tables (hundreds) to achieve good search quality. Bawa et al. proposed the LSH Forest indexing method which represents each hash table by a prefix tree so the number of hash functions per table can be adapted for different approximation distances.

[REF1] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: . These techniques use vector approximations or bounding rectangle approximations to prune search space. Much progress has been made on solving the Approximate Nearest-Neighbor (ANN) problem. The objective is to find points whose distance from the query point is at most 1 + times the exact nearest neighbor’s distance. Due to the limited space, we can not give an extensive review of previous searching and indexing techniques. Please see [7, 15, 12] for some survey. Here, we focus on locality sensitive hashing techniques that are most relevant to our work.

[REF2] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: 2. SIMILARITY SEARCH PROBLEM The problem of similarity search refers to finding objects that have similar characteristics to the query object. When data objects are represented by d-dimensional feature vectors, the goal of similarity search for a given query object q, is to find the K objects that are closest to q according to a distance function in the d-dimensional space. The search quality is measured by the fraction of the nearest K objects we are able to retrieve. In this paper, we also consider the similarity search problem as solving the approximate nearest neighbors problem, where the goal is to find K objects whose distances are within a small factor (1+) of the true K-nearest neighbors’ distances. With this viewpoint, we also measure search quality by comparing the distances to the query for the K objects retrieved to the corresponding distances of the K nearest objects. Our goal is to design a good indexing method for similarity search of large-scale datasets that can achieve high search quality with high time and space efficiency. 3. LSH INDEXING The basic idea of locality sensitive hashing (LSH) is to use hash functions that map similar objects into the same hash buckets with high probability.

[REF3] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: It has been shown in that when the dimensionality exceeds about 10, existing indexing data structures based on space partitioning are slower than the brute-force, linear-scan approach. For high-dimensional similarity search, the best-known indexing method is locality sensitive hashing (LSH) . The basic method uses a family of locality-sensitive hash functions to hash nearby objects in the high-dimensional space into the same bucket. To perform a similarity search, the indexing method hashes a query object into a bucket, uses the data objects in the bucket as the candidate set of the results, and then ranks the candidate objects using the distance measure of the similarity search. To achieve high search accuracy, the LSH method needs to use multiple hash tables to produce a good candidate set. Experimental studies show that this basic LSH method needs over a hundred and sometimes several hundred hash tables to achieve good search accuracy for high-dimensional datasets.

[REF4] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: Current approaches do not satisfy all of these requirements. Previously proposed tree-based indexing methods for KNN search such as R-tree , K-D tree , SR-tree , navigating-nets and cover-tree return accurate results, but they are not time efficient for data with high (intrinsic) dimensionalities. It has been shown in that when the dimensionality exceeds about 10, existing indexing data structures based on space partitioning are slower than the brute-force, linear-scan approach. For high-dimensional similarity search, the best-known indexing method is locality sensitive hashing (LSH)

[REF5] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: Our goal is to design a good indexing method for similarity search of large-scale datasets that can achieve high search quality with high time and space efficiency. 3. LSH INDEXING The basic idea of locality sensitive hashing (LSH) is to use hash functions that map similar objects into the same hash buckets with high probability. Performing a similarity search query on an LSH index consists of two steps: (1) using LSH functions to select “candidate” objects for a given query q, and (2) ranking the candidate objects according to their distances to q. This section provides a brief overview of LSH functions, the basic LSH indexing method and a recently proposed entropy-based LSH indexing method. 3.1 Locality Sensitive Hashing (LSH) The notion of locality sensitive hashing (LSH) was first introduced by Indyk and Motwani in . LSH function families have the property that objects that are close to each other have a higher probability of colliding than objects that are far apart.

[REF6] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF65\_3f1e54ed3bd801766e1897d53a9fc962524dd3c2.pdf Title: Locality-Sensitive Hashing Scheme Based on p-Stable Distributions Chunk of text: This is due to the fact that, in many cases, approximate nearest neighbor is almost as good as the exact one; in particular, if the distance measure accurately captures the notion of user quality, then small differences in the distance should not matter. In fact, in situations when the quality of the approximate nearest neighbor is much worse than the quality of the actual nearest neighbor, then the nearest neighbor problem is unstable, and it is not clear if solving it is at all meaningful [4, 17]. In [19, 14], the authors introduced an approximate high-dimensional similarity search scheme with provably sublinear dependence on the data size. Instead of using tree-like space partitioning, it relied on a new method called locality-sensitive hashing (LSH). The key idea is to hash the points using several hash functions so as to ensure that, for each function, the probability of collision is much higher for objects which are close to each other than for those which are far apart. Then, one can determine near neighbors by hashing the query point and retrieving elements stored in buckets containing that point. In [19, 14] the authors provided such locality-sensitive hash functions for the case when the points live in binary Hamming space f0; 1g d .

[REF7] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: Our evaluation shows that the multi-probe LSH method substantially improves over the basic and entropy-based LSH methods in both space and time efficiency. To achieve over 0.9 recall, the multi-probe LSH method reduces the number of hash tables of the basic LSH method by a factor of 14 to 18 while achieving similar time efficiencies. In comparison with the entropy-based LSH method, multi-probe LSH reduces the space requirement by a factor of 5 to 8 and uses less query time, while achieving the same search quality. We emphasize that our focus in this paper is on improving the space and time efficiency of LSH, already established as an attractive technique for high-dimensional similarity search. We compare our new method to previously proposed LSH methods – a detailed comparison with other indexing techniques is outside the scope of this work. 2. SIMILARITY SEARCH PROBLEM The problem of similarity search refers to finding objects that have similar characteristics to the query object. When data objects are represented by d-dimensional feature vectors, the goal of similarity search for a given query object q, is to find the K objects that are closest to q according to a distance function in the d-dimensional space.

[REF8] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF66\_9ed960374381062d85d3944182a539c1d00f7703.pdf Title: Multi-Probe LSH: Efficient Indexing for High-Dimensional Similarity Search Chunk of text: In practice, it is difficult to generate perturbed queries in a data-independent way and most hashed buckets by the perturbed queries are redundant. The multi-probe LSH method proposed in this paper is inspired by but quite different from the entropybased LSH method. Instead of generating perturbed queries, our method computes a non-overlapped bucket sequence, according to the probability of containing similar objects. 8. CONCLUSIONS This paper presents the multi-probe LSH indexing method for high-dimensional similarity search, which uses carefully derived probing sequences to probe multiple hash buckets in a systematic way. Our experimental results show that the multi-probe LSH method is much more space efficient than the basic LSH and entropy-based LSH methods to achieve desired search accuracy and query time. The multi-probe LSH method reduces the number of hash tables of the basic LSH method by a factor of 14 to 18 and reduces that of the entropy-based approach by a factor of 5 to 8.

[REF9] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Locality\_sensitive\_hashing\_approaches/BIBREF65\_3f1e54ed3bd801766e1897d53a9fc962524dd3c2.pdf Title: Locality-Sensitive Hashing Scheme Based on p-Stable Distributions Chunk of text: This would reduce the memory requirement to l 2, i.e. 60 words per data point. If the data points belong to a high dimensional space (e.g., with 500 dimension or more), then the overhead of maintaining the hash table is not much (around 12% with the optimization above) as compared to storing the points themselves. Thus, the memory overhead of our algorithm is small. 6. CONCLUSIONS In this paper we present a new LSH scheme for the similarity search in high-dimensional spaces. The algorithm is easy to implement, and generalizes to arbitrary lp norm, for p 2 [0; 2℄.

........................................................................................................................................................................................................

Title: Retrieval Architectures and Vector Search - Vector quantisation approaches

Vector quantization is a widely studied technique in information theory that aims to reduce the cardinality of the representation space, particularly for real-valued input data [REF0]. It involves a destructive process of quantization, which has been extensively explored in the literature [REF0]. Various approaches have been proposed, including tree-structured vector quantization (TSVQ), classified vector quantizers, transform vector quantizers, product codes, and multistage vector quantizers [REF2]. These techniques have been applied to different domains such as random processes, speech waveforms, speech models, and images [REF3].

In the context of neural information retrieval, vector quantization plays a crucial role in nearest neighbor search, where the distances between the query vector and the database vectors are compared [REF1]. This comparison is often based on the quantization indices of the vectors, inspired by source coding techniques [REF1]. The use of quantized codes allows for efficient computation of approximate Euclidean distances between vectors [REF4]. Two methods, symmetric and asymmetric, have been proposed for this purpose [REF4].

The trade-offs between memory usage and search accuracy are important considerations in vector quantization. The product quantizer, which is parametrized by the number of subvectors and the number of quantizers per subvector, offers a trade-off between code length and search quality [REF6]. The choice of these parameters affects the performance of the retrieval system, and finding the optimal balance is crucial [REF6].

While lattice quantizers offer better quantization properties for uniform vector distributions, they often perform worse than k-means in indexing tasks [REF7]. In the context of nearest neighbor search, product quantizers have shown advantages over other methods, such as higher number of possible distances and estimation of expected squared distance [REF7]. These advantages make product quantizers a promising approach for neural information retrieval.

Evaluation of vector quantization approaches in neural information retrieval involves comparing them with state-of-the-art methods. Spectral hashing, Hamming embedding, and FLANN are among the existing techniques that have been used for comparison [REF5, REF9]. Performance measures such as precision and recall@100 are commonly used to evaluate the retrieval accuracy [REF6]. The evaluation also includes analyzing the impact of different parameters on the performance of the retrieval system [REF5].

In summary, vector quantization approaches, particularly product quantizers, play a significant role in retrieval architectures and vector search in neural information retrieval. These approaches involve quantizing vectors and comparing them based on their quantization indices. The choice of parameters in vector quantization affects the trade-off between memory usage and search accuracy. Evaluation of these approaches involves comparing them with state-of-the-art methods and analyzing their performance under different parameter settings.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: Section III presents our approach for NN search and Section IV introduces the structure used to avoid exhaustive search. An evaluation of the parameters of our approach and a comparison with the state of the art is given in Section V. II. BACKGROUND: QUANTIZATION, PRODUCT QUANTIZER A large body of literature is available on vector quantization, see for a survey. In this section, we restrict our presentation to the notations and concepts used in the rest of the paper. A. Vector quantization Quantization is a destructive process which has been extensively studied in information theory . Its purpose is to reduce the cardinality of the representation space, in particular when the input data is real-valued.

[REF1] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: (right). The mean squared error on the distance is on average bounded by the quantization error. k ∗ = 256 and m = 8 is often a reasonable choice. III. SEARCHING WITH QUANTIZATION Nearest neighbor search depends on the distances between the query vector and the database vectors, or equivalently the squared distances. The method introduced in this section compares the vectors based on their quantization indices, in the spirit of source coding techniques.

[REF2] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF67\_c564aa7639a08c280423489e52b6e32055c9aa7f.pdf Title: Vector Quantization and Signal Compression Chunk of text: Included are tree-structured vector quantization (TSVQ), classified vector quantizers, transform vector quantizers, product codes such as gain/shape and mean-residual vector quantizers, and multistage vector quantizers. Also covered are fast search algorithms for codebook searching nonlinear interpolative coding, and hierarchical coding. Chapters 13 and 14 consider vector quantizers with memory, sometimes called recursive vector quantizers or feedback vector quantizers. Chapter 13 treats the extension of predictive quantization to vectors, predictive vector quantization (PVQ). Here vector predictors are used to form a prediction residual of the original input vector and the resulting residual is quantized. This chapter builds on the linear prediction theory of Chapter 4 and develops some vector extensions for more sophisticated systems. Chapter 14 treats finite-state vector quantization (FSVQ) wherein the encoder and decoder are finite-state machines.

[REF3] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF67\_c564aa7639a08c280423489e52b6e32055c9aa7f.pdf Title: Vector Quantization and Signal Compression Chunk of text: Chapters 10 and 11 extend the fundamentals of scalar quantization of Chapters 5 and 6 to vectors. Chapter 10 provides the motivation, definitions, properies, structures, and figures of merit of vector quantization. Chapter 11 develops the basic optimality properties for vector quantizers and extends the Lloyd clustering algorithm to vectors. A variety of design examples to random processes, speech waveforms, speech models, and images are described and pursued through the subsequent chapters. Chapter 12 considers the shortcomings in terms of complexity and memory of simple memoryless, unconstrained vector quantizers and provides a variety of constrained coding schemes that provide reduced complexity and better performance in trade for a tolerable loss of optimality. Included are tree-structured vector quantization (TSVQ), classified vector quantizers, transform vector quantizers, product codes such as gain/shape and mean-residual vector quantizers, and multistage vector quantizers. Also covered are fast search algorithms for codebook searching nonlinear interpolative coding, and hierarchical coding.

[REF4] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: Nearest neighbor search depends on the distances between the query vector and the database vectors, or equivalently the squared distances. The method introduced in this section compares the vectors based on their quantization indices, in the spirit of source coding techniques. We first explain how the product quantizer properties are used to compute the distances. Then we provide a statistical bound on the distance estimation error, and propose a refined estimator for the squared Euclidean distance. A. Computing distances using quantized codes Let us consider the query vector x and a database vector y. We propose two methods to compute an approximate Euclidean distance between these vectors,5 a symmetric and a asymmetric one. See Figure 2 for an illustration.

[REF5] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: Compared with ADC, the additional step of quantizing x to qc(x) consists in computing k ′ distances between Ddimensional vectors. Assuming that the inverted lists are balanced, about n × w/k′ entries have to be parsed. Therefore, the search is significantly faster than ADC, as shown in the next section. V. EVALUATION OF NN SEARCH In this section, we first present the datasets used for the evaluation3 . We then analyze the impact of the parameters for SDC, ADC and IVFADC. Our approach is compared to three state-of-the-art methods: spectral hashing , Hamming embedding and FLANN

[REF6] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: This measure indicates the fraction of queries for which the nearest neighbor is retrieved correctly, if a short-list of R vectors is verified using Euclidean distances. Furthermore, the curve obtained by varying R corresponds to the distribution function of the ranks, and the point R=1 corresponds to the “precision” measure used in to evaluate ANN methods. In practice, we are often interested in retrieving the K nearest neighbors (K > 1) and not only the nearest neighbor. We do not include these measures in the paper, as we observed that the conclusions for K=1 remain valid for K > 1. B. Memory vs search accuracy: trade-offs The product quantizer is parametrized by the number of subvectors m and the number of quantizers per subvector k ∗ , producing a code of length m × log2 k ∗ . Figure 6 shows the trade-off between code length and search quality for our SIFT descriptor dataset. The quality is measured for recall@100 for the ADC and SDC estimators, for m ∈ {1, 2, 4, 8, 16} and k ∗ ∈ {2 4 , 2 6 , 2 8 , 2 10 , 2 12}.

[REF7] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: Lattice quantizers offer better quantization properties for uniform vector distributions, but this condition is rarely satisfied by real world vectors. In practice, these quantizers perform significantly worse than k-means in indexing tasks . In this paper, we focus on product quantizers. To our knowledge, such a semi-structured quantizer has never been considered in any nearest neighbor search method. The advantages of our method are twofold. First, the number of possible distances is significantly higher than for competing Hamming embedding methods , , , as the Hamming space used in these techniques allows for a few distinct distances only. Second, as a byproduct of the method, we get an estimation of the expected squared distance, which is required for ε-radius search or for using Lowe’s distance ratio criterion .

[REF8] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: The search consists in comparing the Hamming distances between the database signatures and the query vector signature. This approach was shown to outperform the restricted Boltzmann machine of . We have used the publicly available code. We also compare to the Hamming embedding (HE) method of , which also maps vectors to binary signatures. Similar to IVFADC, HE uses an inverted file, which avoids comparing to all the database elements. Figures 8 and 9 show, respectively for the SIFT and the GIST datasets, the rank repartition of the nearest neighbors when using a signature of size 64 bits. For our product quantizer we have used m

[REF9] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Vector\_quantisation\_approaches/BIBREF68\_4748d22348e72e6e06c2476486afddbc76e5eca7.pdf Title: Product Quantization for Nearest Neighbor Search Chunk of text: A comparison with the state of the art shows that our approach outperforms existing techniques, in particular spectral hashing , Hamming embedding and FLANN . Our paper is organized as follows. Section II introduces the notations for quantization as well as the product quantizer used by our method. Section III presents our approach for NN search and Section IV introduces the structure used to avoid exhaustive search. An evaluation of the parameters of our approach and a comparison with the state of the art is given in Section V. II.

........................................................................................................................................................................................................

Title: Retrieval Architectures and Vector Search - Graph approaches

Graph-based approaches have gained significant attention in the field of neural information retrieval due to their ability to capture complex relationships and dependencies among data points. In particular, graph-based retrieval architectures have shown promise in addressing the challenges of efficient similarity search and approximate nearest neighbor (ANN) graph construction [REF0] [REF1].

One common problem in similarity search is the k-nearest neighbor search (k-NNS), where the goal is to find the k closest objects to a given query based on a distance function [REF1]. The naïve approach of calculating the distance between the query and every element in the dataset is computationally expensive and not feasible for large-scale datasets [REF4] [REF5]. To overcome this limitation, graph-based retrieval architectures represent the dataset as a graph, where each object is associated with a vertex [REF1]. Searching for the closest elements to the query then becomes a search for vertices in the graph [REF1]. This approach enables the development of decentralized similarity search-oriented storage systems, where physical data location is independent of content [REF1].

In the context of approximate nearest neighbor graph construction, efficient methods based on prefix-filtering have been developed [REF0] [REF3]. These methods construct an ǫ-NN (epsilon-nearest neighbor) graph, which establishes an edge between all pairs of points whose similarity is above a certain threshold ǫ [REF0] [REF3]. However, these methods are efficient only for a very tight similarity threshold, resulting in a sparse and disconnected graph [REF0]. To address this limitation, recent research has focused on developing graph-based retrieval architectures that can construct approximate k-nearest neighbor (K-NN) graphs with arbitrary similarity measures [REF0]. One such method is NN-Descent, which has demonstrated excellent accuracy and speed in approximate K-NN graph construction [REF0].

Graph-based retrieval architectures also leverage the use of locality-sensitive hashing (LSH) for approximate K-NN search in high-dimensional spaces [REF8]. LSH is a promising method that uses multiple hash tables to build an LSH index, which is then used to run K-NN queries [REF8]. Optimizations such as using bit vectors to avoid redundant evaluations of the same points have been proposed to improve the efficiency of LSH-based retrieval architectures [REF8].

The performance of graph-based retrieval architectures has been evaluated on various datasets, demonstrating their effectiveness in achieving high recall and search speed [REF2] [REF6] [REF7]. For instance, experiments have shown that only a small fraction of the database needs to be evaluated to achieve high recall, making the search virtually exact [REF2] [REF6] [REF7]. Additionally, these architectures have shown scalability and parallelizability, enabling their application to large-scale datasets without an explicit graph structure [REF0] [REF2].

In conclusion, graph-based retrieval architectures offer promising solutions for efficient similarity search and approximate nearest neighbor graph construction. These architectures leverage the power of graphs to capture complex relationships among data points and have demonstrated excellent accuracy and speed in various experiments. Further research in this area can explore novel graph-based approaches and optimizations to enhance the performance and scalability of neural information retrieval systems.

References:

[REF0] [13, 17]

[REF1] [15]

[REF2] [REF7]

[REF3] [REF5]

[REF4] [REF6]

[REF5] [REF8]

[REF6] [REF9]

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF69\_f17c6e164ccc7ec1ad91b3fbbafe8f84664e9803.pdf Title: Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures Chunk of text: [13, 17]. As we have shown with experiments, it is hard for LSH to achieve high recall, and designing an affective hash function for a new similarity measure is nontrivial. In the text retrieval community, efficient methods based on prefix-filtering are developed for the ǫ-NN graph construction [2, 22, 21], a different kind of nearest neighbor graph which establishes an edge between all pairs of points whose similarity is above ǫ. The problem is that such methods are only efficient for a very tight similarity threshold, corresponding to a very sparse and disconnected graph. 6. CONCLUSION We presented NN-Descent , a simple and efficient method for approximate K-Nearest Neighbor graph construction with arbitrary similarity measures, and demonstrated its excellent accuracy and speed with extensive experimental study. Our method has a low empirical complexity of O(n 1.14) (on various tested datasets) and can be easily parallelized, potentially enabling the application of existing graph and network analysis methods to large-scaled dataset without an explicit graph structure.

[REF1] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF70\_c197ecb6a6987667cadcb498136989af1827cce0.pdf Title: Approximate Nearest Neighbor Algorithm based on Navigable Small World Graphs Chunk of text: Information Systems 45 (2014) 61–68Author's personal copy document retrieval ) that require the similarity search rather than just exact matching. The k-nearest neighbor search (k-NNS) problem is a mathematical formalization for similarity search. It is defined as follows: we need to find the set of k closest objects PDX from a finite set of objects XDD to a given query qAD, where D is the set of all possible objects (the data domain). Closeness or proximity of two objects o′; o″AD is defined as a distance function δðo′; o″Þ. A naïve solution for the k-NNS problem is to calculate the distance function δ between q and every element from X. This leads to linear search time complexity, which is much worse than the scalability of structures for exact match search, and makes the naïve version of k-NNS almost impossible to use for large size datasets. We suggest a solution for the nearest neighbor search problem: a data structure represented by a graph GðV; EÞ, where every object oi from X is uniquely associated with a vertex vi from V. Searching for the closest elements to the query q from the data set X takes the form of searching for a vertices in the graph G. This gives an opportunity for building decentralized similarity search oriented storage systems where physical data location does not depend on the content because every data object can be placed on an arbitrary physical machine and can be connected with others by links like in p2p systems.

[REF2] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF70\_c197ecb6a6987667cadcb498136989af1827cce0.pdf Title: Approximate Nearest Neighbor Algorithm based on Navigable Small World Graphs Chunk of text: k¼30 nearest neighbors were found during a search. 100 thousands different (other) elements from the dataset were used as queries. Construction of the structure was done in parallel by 16 threads and took about 2 h. Since for optimal f30–40 (effective dimensionality 10–13) the algorithm achieves high recall even at a single search, we have compared the recall error (one minus recall) instead of recall (see in Fig. 6 the recall error versus fraction of the visited elements in a logarithmic scale). The achieved results are even slightly better than the expected exponential decrease. Only 0.031% of the database needed to be evaluated to get 0.999 recall, which makes the search virtually exact. In terms of throughput, at m¼1 with recall 0.92 about 2800 searches per second can be done in parallel on our 12-core test system.

[REF3] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF69\_f17c6e164ccc7ec1ad91b3fbbafe8f84664e9803.pdf Title: Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures Chunk of text: is a promising method for approximate KNN search. Such hash functions have been designed for a range of different similarity measures, including hamming distance , lp with p ∈ (0, 2] , cosine similarity , etc. However, the computational cost remains high for achieving accurate approximation, and designing an effective hash function for a new similarity measure is non-trivial. In the text retrieval community, methods based on prefixfiltering have been developed for ǫ-NN graph construction, a.k.a. all pairs similarity search or similarity join [2, 22, 21]. An ǫ-NN graph is different from a K-NNG in that undirected edges are established between all pairs of points with a similarity above ǫ.

[REF4] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF72\_699a2e3b653c69aff5cf7a9923793b974f8ca164.pdf Title: Efficient and Robust Approximate Nearest Neighbor Search using Hierarchical Navigable Small World Graphs Chunk of text: INTRODUCTION onstantly growing amount of the available information resources has led to high demand in scalable and efficient similarity search data structures. One of the generally used approaches for information search is the K-Nearest Neighbor Search (K-NNS). The K-NNS assumes you have a defined distance function between the data elements and aims at finding the K elements from the dataset which minimize the distance to a given query. Such algorithms are used in many applications, such as non-parametric machine learning algorithms, image features matching in large scale databases and semantic document retrieval . A naïve approach to K-NNS is to compute the distances between the query and every element in the dataset and select the elements with minimal distance. Unfortunately, the complexity of the naïve approach scales linearly with the number of stored elements making it infeasible for large-scale datasets.

[REF5] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF72\_699a2e3b653c69aff5cf7a9923793b974f8ca164.pdf Title: Efficient and Robust Approximate Nearest Neighbor Search using Hierarchical Navigable Small World Graphs Chunk of text: A naïve approach to K-NNS is to compute the distances between the query and every element in the dataset and select the elements with minimal distance. Unfortunately, the complexity of the naïve approach scales linearly with the number of stored elements making it infeasible for large-scale datasets. This has led to a high interest in development of fast and scalable KNNS algorithms. Exact solutions for K-NNS [3-5] may offer a substantial search speedup only in case of relatively low dimensional data due to “curse of dimensionality”. To overcome this problem a concept of Approximate Nearest Neighbors Search (K-ANNS) was proposed, which relaxes the condition of the exact search by allowing a small number of errors. The quality of an inexact search (the recall) is defined as the ratio between the number of found true nearest neighbors and K. The most popular K-ANNS solutions are based on approximated versions of tree algorithms

[REF6] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF70\_c197ecb6a6987667cadcb498136989af1827cce0.pdf Title: Approximate Nearest Neighbor Algorithm based on Navigable Small World Graphs Chunk of text: Only 0.031% of the database needed to be evaluated to get 0.999 recall, which makes the search virtually exact. In terms of throughput, at m¼1 with recall 0.92 about 2800 searches per second can be done in parallel on our 12-core test system. The inset of the Fig. 6 shows logarithmic rise of number of evaluated elements for a single 0.999 recall search with the growth of the dataset size. See the Fig. 7 for the comparison with the data for NAPP, with K¼7 the parameter . Values of s in NAPP were selected to get best recall at fixed fraction of visited elements. Our algorithm is very effective at big dataset size, especially in case of high recall, requiring more that hundred time less metric computation at a recall of 0.999. The comparison to Ordering Permutation index at low(104 ) number of points but high dimensionality (d¼1024), which means that the small world navigation properties do not play a critical role, showed that our algorithm yields in performance(about 65% database elements visited for our algorithm get 0.9 recall versus 42% for the OP).

[REF7] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF70\_c197ecb6a6987667cadcb498136989af1827cce0.pdf Title: Approximate Nearest Neighbor Algorithm based on Navigable Small World Graphs Chunk of text: Only 0.03% percent of the 10 million 208-dimensional CoPHiR dataset is needed to be evaluated to achieve 0.999 recall (virtually exact search). For recall 0.93 processing speed 2800 queries/s can achieved on a single server node. Fig. 6. Average fraction of visited elements within a single k-NN-search vs recall error for 10 M 208 dimensional vectors from CoPHiR database. The inset shows logarithmic rise of distance calculations to get 0.999 recall (vertical) with the dataset size (horizontal). Fig. 7.

[REF8] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF69\_f17c6e164ccc7ec1ad91b3fbbafe8f84664e9803.pdf Title: Efficient K-Nearest Neighbor Graph Construction for Generic Similarity Measures Chunk of text: 4.2.2 Locality Sensitive Hashing LSH is a promising method for approximate K-NN search in high dimensional spaces. We use LSH for offline K-NNG construction by building an LSH index (with multiple hash tables) and then running a K-NN query for each object. We use plain LSH rather than the more recent MultiProbing LSH in this evaluation as the latter is mainly to reduce space cost, but could slightly raise scan rate to achieve the same recall. We make the following optimizations to the original LSH method to better suit the K-NNG construction task: • For each query, we use a bit vector to record the objects that have been compared, so if the same points are seen in another hash table, they are not evaluated again. Dataset LSH Ours recall scan rate recall scan rate Corel 0.906 0.004 0.995 0.004 Audio 0.615 0.047 0.969 0.045 Shape 0.925 0.076 0.994 0.075

[REF9] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Graph\_approaches/BIBREF72\_699a2e3b653c69aff5cf7a9923793b974f8ca164.pdf Title: Efficient and Robust Approximate Nearest Neighbor Search using Hierarchical Navigable Small World Graphs Chunk of text: 15 presents the scaling of the query time vs the dataset size for Hierarchical NSW. Note that the scaling deviates from the pure logarithm, possibly due to relatively high dimensionality of the dataset. 6 DISCUSSION By using structure decomposition of navigable small world graphs together with the smart neighbor selection heuristic the proposed Hierarchical NSW approach overcomes several important problems of the basic NSW structure advancing the state-of–the-art in K-ANN search. Hierarchical NSW offers an excellent performance and is a clear leader on a large variety of the datasets, surpassing the opensource rivals by a large margin in case of high dimensional data. Even for the datasets where the previous algorithm (NSW) has lost by orders of magnitude, Hierarchical NSW was able to come first. Hierarchical NSW supports continuous incremental indexing and can also be used as an efficient method for getting approximations of the k-NN and relative neighborhood graphs, which are byproducts of the index construction. Robustness of the approach is a strong feature which makes it very attractive for practical applications.

........................................................................................................................................................................................................

Title: Retrieval Architectures and Vector Search - Optimisations

Most neural ranking approaches have traditionally focused on re-ranking documents identified by a classical inverted index using relevance models such as BM25 in a multi-stage ranking architecture [REF0]. However, relying solely on lexical matching models based on an inverted index may not effectively identify contextually related candidate documents that would receive high scores from a neural ranking model. To address this limitation, dense retrieval approaches have gained increasing interest [REF0]. In dense retrieval, documents are encoded as vectors during indexing, and queries are encoded as vectors during query processing. The top-ranked documents for a given query are then computed by identifying the most similar document embeddings to the query embeddings using a nearest neighbor search procedure [REF0].

Nearest neighbor search with single representations has been shown to be efficient but less effective than using multiple representations [REF0]. Khattab and Zaharia proposed a two-stage dense retrieval cascading approach that leverages multiple representations [REF6]. In the first stage, an approximate nearest neighbor (ANN) search is performed to retrieve a set of candidate documents, maximizing the recall of the retrieved set. In the second stage, accurate scores are computed for the candidate documents from the first stage to determine the final top documents [REF6].

To optimize the efficiency of dense retrieval, various optimizations have been explored. One optimization is the use of quantized document embeddings, which allows for fast searching while sacrificing some accuracy in the approximate similarity scores [REF9]. Another optimization is the partitioning of the index into multiple partitions, which enables parallel processing and reduces the search space [REF1]. Additionally, the use of hardware accelerators, such as GPUs, can significantly improve the retrieval speed [REF1].

Comparisons with other retrieval models have also been conducted to evaluate the effectiveness and efficiency of dense retrieval. Existing neural retrieval models, such as brute-force dense retrieval models and late-interaction models, have shown to be more effective than traditional bag-of-words models but significantly increase the index size [REF2]. Complex end-to-end neural retrieval models achieve better ranking performance but at the cost of higher query latency due to their complex model architecture [REF3]. Therefore, a trade-off between effectiveness and latency needs to be considered when comparing dense retrieval with other retrieval models [REF3].

In summary, retrieval architectures for neural information retrieval have evolved from re-ranking documents identified by an inverted index to dense retrieval approaches that utilize document and query embeddings. Optimizations such as multiple representations, quantization, and hardware acceleration have been explored to improve the efficiency of dense retrieval. Comparisons with other retrieval models highlight the trade-offs between effectiveness and latency. These advancements in retrieval architectures and vector search optimizations contribute to the development of more efficient and effective neural information retrieval systems.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF77\_c3cf35677834fb535d3bc7cf8d375366df4b1397.pdf Title: Query Embedding Pruning for Dense Retrieval Chunk of text: Most neural ranking approaches have been used to by re-rank the documents identified by a classical inverted index using relevance models such as BM25, in a multi-stage ranking architecture [6, 13]. However, lexical matching models relying solely on an inverted index may not identify the contextually related candidate documents that would have been highly scored by an effective neural ranking model. Instead, by utilising documents encoded as vectors at indexing time and queries encoded as vectors at query processing time, dense retrieval approaches [5, 17] are of growing interest. In dense retrieval, the topranked documents for a given query are computed by identifying the most similar document embeddings to the given query embeddings, employing a nearest neighbour search procedure. Nearest neighbour search with single representations has been shown to be efficient, but less effective than multiple representations . On the other hand, when multiple representations are exploited, as pioneered by Khattab and Zaharia

[REF1] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF55\_60b8ad6177230ad5402af409a6edb5af441baeb4.pdf Title: ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT Chunk of text: For our faiss index, we set the number of partitions to P =2,000, and search the nearest p = 10 to each query embedding to retrieve k 0 = k = 1000 document vectors per query embedding. We divide each embedding into s = 16 sub-vectors, each encoded using one byte. To represent the index used for the second stage of our end-to-end retrieval procedure, we use 16-bit values per dimension. 4.1.3 Hardware & Time Measurements. To evaluate the latency of neural re-ranking models in §4.2, we use a single Tesla V100 GPU that has 32 GiBs of memory on a server with two Intel Xeon Gold 6132 CPUs, each with 14 physical cores (24 hyperthreads), and 469 GiBs of RAM. For the mostly CPU-based retrieval experiments in §4.3 and the indexing experiments in §4.5, we use another server with the same CPU and system memory specications but which has four Titan V GPUs aached, each with 12 GiBs of memory.

[REF2] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF75\_d6f83c915565f575e55fdce0424f65fe192af218.pdf Title: Jointly Optimizing Query Encoder and Product Quantization to Improve Retrieval Performance Chunk of text: 5.2 Comparison with Retrieval Models This section uses existing retrieval models as baselines to answer RQ2. We firstly compare JPQ with baselines in general, and then separately compare JPQ with different types of baselines in detail. 5.2.1 Overall Comparison. We summarize the ranking performance, index size, and latency of representative retrieval models in Figure 1, Figure 6, and Table 2. According to the figures, although existing neural retrieval models, i.e., brute-force DR models and late-interaction models, are more effective than BoW models, they significantly increase the index size by several orders of magnitude. When the indexes of brute-force DR models are compressed by LSH or OPQ , the retrieval effectiveness is severely hurt. Therefore, the results seem to imply that large index sizes are necessary for high-quality ranking.

[REF3] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF76\_91429255eefe48ad140ccfaf6aa1e6be11a72a53.pdf Title: Learning Discrete Representations via Constrained Clustering for Effective and Efficient Dense Retrieval Chunk of text: The results demonstrate the time efficiency of RepCONC. 5.2.2 Comparison with Complex End-to-End Retrievers. This section compares RepCONC with some complex (slow) end-to-end neural retrieval models. These models achieve better ranking performance with much higher query latency because of their complex model architecture. In consideration of fair comparison, we add a reranking stage to RepCONC and compare them in terms of effectiveness-latency tradeoff. The reranking modelsWSDM’22, February 21-25, 2022, Phoenix, Arizona Zhan, et al. 10 0 10 1 QPS 0.34 0.36 0.38 0.40 MRR@10 RepCONC-IVF+Rerank ColBERT COIL-Hard COIL uniCOIL DeepImpact Figure 6: Comparison with complex (slow) end-to-end retrieval models in terms of effectiveness-latency tradeoff on MS MARCO Passage Ranking. The search is performed on CPU with one thread.

[REF4] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF51\_c9b8593db099869fe7254aa1fa53f3c9073b0176.pdf Title: Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval Chunk of text: 7 RELATED WORK In early research on neural information retrieval (Neu-IR) (Mitra et al., 2018), a common belief was that the interaction models, those that specifically handle term level matches, are more effective though more expensive (Guo et al., 2016; Xiong et al., 2017; Nogueira & Cho, 2019). Many techniques are developed to reduce their cost, for example, distillation (Gao et al., 2020a) and caching (Humeau et al., 2020; Khattab & Zaharia, 2020; MacAvaney et al., 2020). ANCE shows that a properly trained representation-based BERT-Siamese is in fact as effective as the interaction-based BERT ranker. This finding will motivate many new research explorations in Neu-IR. Deep learning has been used to improve various components of sparse retrieval, for example, term weighting (Dai & Callan, 2019b), query expansion (Zheng et al., 2020), and document expansion (Nogueira et al., 2019). Dense Retrieval chooses a different path and conducts retrieval purely in the embedding space via ANN search (Lee et al., 2019; Chang et al., 2020; Karpukhin et al., 2020; Luan et al., 2020). This work demonstrates that a simple dense retrieval system can achieve SOTA accuracy, while also behaves dramatically different from classic retrieval.

[REF5] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF52\_7b577ba0e4230b2ac58d297b3d2cfc3d2f1aaace.pdf Title: Optimizing Dense Retrieval Model Training with Hard Negatives Chunk of text: To better deduce the users’ search intent and retrieve relevant items, the ranking algorithms are expected to conduct semantic matching between queries and documents , which is a challenging problem. In recent years, with the development of deep learning [6, 20, 28], especially representation learning techniques , many researchers have turned to the Dense Retrieval (DR) model to solve the semantic matching problem [10, 15, 18, 22]. In essence, DR attempts to encode queries and documents into low-dimension embeddings to better abstract their semantic meanings. With the learned embeddings, document index can be constructed and the query embedding can be adopted to perform efficient similarity search for online ranking. Previous studies showed that DR models achieve promising results on many IR-related tasks [7, 10, 15]. However, there are some unsolved but essential problems related to DR’s effectiveness and training efficiency1 . Firstly, though 1We focus on the training efficiency because the efficiency in the inference process is guaranteed by the maximum inner product search algorithms[14, 27]. arXiv:2104.08051v1

[REF6] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF78\_2408c965a3855a6b66d128195c783d76e2e939da.pdf Title: On Approximate Nearest Neighbour Selection for Multi-Stage Dense Retrieval Chunk of text: When single representations for both documents and queries are used, exact nearest neighbour (NN) search data structures and algorithms can be exploited [8, 22]. While NN search on single representations has been shown to be efficient, it is less effective than NN search on multiple representations . On the other hand, to scale to number of required vectors, multiple representations require approximate nearest neighbour (ANN) data structures and algorithms. Hence, Khattab and Zaharia propose a two-stage dense retrieval cascading approach. The first stage performs the ANN search to retrieve a set of candidate documents, maximising the recall of the retrieved set. The second stage computes accurate scores for the first-stage candidate documents, to return the final top documents.

[REF7] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF74\_28336cbf2ee3e8fca6b173c91c5ca9628ba1fa4a.pdf Title: Joint Learning Deep Retrieval Model and Product Quantization based Embedding Index Chunk of text: Introduction Various types of indexes play an indispensable role in modern computational systems to enable fast information retrieval. As a traditional one, inverted index has been widely used in web search, e-commerce search, recommendation and advertising in the past few decades. Recently, with the advent of the deep learning era, embedding indexes [6, 15], which embed user/query and item in a latent vector space, show excellent performance in many industrial retrieval systems [14, 20, 26]. Embedding index enjoys several appealing advantages: a) the embeddings can be learned to optimize downstream retrieval task of interests, and b) efficient algorithms for maximum inner product search (MIPS) or approximate nearest neighbors (ANN), such as LSH , Annoy and state-of-the-art product quantization (PQ) based approaches [11, 17, 18], can be leveraged to retrieve items in a few milliseconds.

[REF8] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF55\_60b8ad6177230ad5402af409a6edb5af441baeb4.pdf Title: ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT Chunk of text: introduced SNRM, a representationfocused IR model that encodes each query and each document as a single, sparse high-dimensional vector of “latent terms”. By producing a sparse-vector representation for each document, SNRM is able to use a traditional IR inverted index for representing documents, allowing fast end-to-end retrieval. Despite highly promising results and insights, SNRM’s eectiveness is substantially outperformed by the state of the art on the datasets with which it was evaluated (e.g., see [18, 38]). While SNRM employs sparsity to allow using inverted indexes, we relax this assumption and compare a (dense) BERT-based representation-focused model against our late-interaction ColBERT in our ablation experiments in §4.4. For a detailed overview of existing neural ranking models, we refer the readers to two recent surveys of the literature [8, 21]. Language Model Pretraining for IR. Recent work in NLU emphasizes the importance pre-training language representation models in an unsupervised fashion before subsequently ne-tuning them on downstream tasks.

[REF9] - paperID: ./papers\_pdf/paper\_section/Retrieval\_Architectures\_and\_Vector\_Search-Optimisations/BIBREF77\_c3cf35677834fb535d3bc7cf8d375366df4b1397.pdf Title: Query Embedding Pruning for Dense Retrieval Chunk of text: With ANN, the document embeddings are stored in a quantised form, suitable for fast searching. However, the approximate similarity scores between these compressed embeddings are inaccurate, and hence are not used for computing the final top documents. Indeed, ANN search computes, for each query embedding 1 In current practice , queries are augmented up to 32 query token embeddings. Query text Query embeddings Retrieved documents Top documents fQ |q| 1 2 Approximate nearest neighbour Learned query search representation D1(k0 ) D2(k0 ) D|q|(k0 ) k Quantised document embeddings Reranker Document embeddings Figure 1: Dense retrieval architecture. 𝜙𝑖 , the set Ψ(𝜙𝑖 , 𝑘′ ) of the 𝑘 ′ document embeddings most similar to 𝜙𝑖 according to some approximate distance; then, these document embeddings are mapped back to their documents 𝐷𝑖(𝑘 ′ ): 𝐷𝑖(𝑘 ′ )

........................................................................................................................................................................................................

Title: Learned Sparse Retrieval - Document expansion learning

Several approaches have been proposed to enhance traditional sparse Bag-of-Words (BoW) representations using Pre-trained Language Models (PLMs) [REF0]. For instance, Dai introduced DeepCT, which estimates term weights by considering contextualized information [REF0]. This work was later extended to generate document-level term weights [REF0]. Another approach, Doc2query, aims to expand document content by "translating" potential queries [REF0]. Doc2query has shown significant improvement compared to the traditional BM25 method [REF0]. However, a key distinction between these methods and our work is that DeepCT and Doc2query train an auxiliary intermediate model to refine sparse representations, while our approach, SparTerm, directly learns sparse representations within the entire vocabulary [REF0].

SparTerm employs a model architecture and training strategy that enables the learning of sparse representations [REF1]. The Gating Controller parameters are fixed, and SparTerm is fine-tuned jointly for 100k iterations using the Adam optimizer with a learning rate of 2 × 10−5 [REF1]. To ensure sparsity, a threshold of 0.7 is set in the Binarizer [REF1]. Notably, our model is not fine-tuned on the training set of the document retrieval dataset but instead utilizes the model trained on the passage retrieval dataset for document ranking [REF1].

Document expansion techniques have shown promise in improving retrieval effectiveness [REF2]. By enriching document representations prior to indexing, document expansion can yield noticeable improvements over traditional "bag of words" baselines without the need for expensive neural network inference at retrieval time [REF2]. While document expansion has been less popular in information retrieval (IR) research due to the need for re-indexing, recent advancements in neural ranking models offer solutions to the vocabulary mismatch problem [REF3]. These models leverage continuous word representations and the ability to learn highly non-linear relevance models [REF3].

SparTerm demonstrates the potential for improving the ranking performance of term-based representations while maintaining the interpretability and efficiency of BoW methods [REF4]. In evaluations on the MSMARCO dataset, SparTerm outperforms previous sparse models based on comparable PLM sizes [REF4]. Notably, SparTerm even outperforms Doc2Query-T5, which is based on a larger pre-trained model and a larger pre-training corpus [REF4]. Furthermore, empirical analysis reveals insights into how deep knowledge from PLMs can be transferred to sparse representation learning [REF4].

In summary, learned sparse retrieval through document expansion learning has shown promise in enhancing traditional sparse representations for information retrieval tasks. By leveraging PLMs and training strategies like SparTerm, researchers have achieved notable improvements in retrieval effectiveness and gained insights into the transferability of deep knowledge from PLMs to sparse methods [REF0][REF1][REF2][REF3][REF4].

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: 2.3 PLMs for sparse text retrieval Several PLM-based models have emerged to improve the traditional sparse BoW representations. Dai proposed DeepCT to estimate a term’s weight considering its contextualized information, and this work was later extended to generate document-level term weights . Another work Doc2query tries to “translate” potential queries to expand document content, which also shows a large improvement compared to the traditional BM25 method. The biggest difference between our work and these two methods is that DeepCT and Doc2Query train an auxiliary intermediate model to help refine the sparse representations, while SparTerm is desinged to directly learn sparse representations within the whole vocabulary. 3 SPARSE REPRESENTATION LEARNING This section presents the model architecture of SparTerm and the corresponding training strategy.

[REF1] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: Then we fix the parameters of the Gating Controller and fine-tune our SparTerm jointly for 100k iterations. We use Adam optimizer with the learning rate 2 × 10−5 . To ensure the sparsity, the threshold in the Binarizer in Equation (4) is set to 0.7. We do not fine-tune our model on the training set of document retrieval dataset but just use the model trained on the passage retrieval dataset for the document ranking. 4.3 Baselines and Experimental Settings We compare our model with the following strong baselines which are all methods based on sparse representation . The former two focus on re-weighting while the latter two focus on document expansion: • BM25 is a bag-of-words retrieval models with frequencybased signals to estimate the weights of terms in a text. • DeepCT is a deep contextualized term weighting model which maps the BERT’s representations to term weightings for retrieval.

[REF2] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF82\_b092b6b843e9421bf42bf96f57ed4658a3e0bdf7.pdf Title: Document Expansion by Query Prediction Chunk of text: On the recent MS MARCO dataset (Bajaj et al., 2016), our approach is competitive with the best results on the official leaderboard, and we report the best-known results on TREC CAR (Dietz et al., 2017). We further show that document expansion is more effective than query expansion on these two datasets. We accomplish this with relatively simple models using existing open-source toolkits, which allows easy replication of our results. Document expansion arXiv:1904.08375v2 [cs.IR] 25 Sep 20192 also presents another major advantage, since the enrichment is performed prior to indexing: Although retrieved output can be further re-ranked using a neural model to greatly enhance effectiveness, the output can also be returned as-is. These results already yield a noticeable improvement in effectiveness over a “bag of words” baseline without the need to apply expensive and slow neural network inference at retrieval time.

[REF3] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF82\_b092b6b843e9421bf42bf96f57ed4658a3e0bdf7.pdf Title: Document Expansion by Query Prediction Chunk of text: These techniques share in their focus on enhancing query representations to better match documents. In this work, we adopt the alternative approach of enriching document representations (Tao et al., 2006; Pickens et al., 2010; Efron et al., 2012), which works particularly well for speech (Singhal and Pereira, 1999) and multi-lingual retrieval, where terms are noisy. Document expansion techniques have been less popular with IR researchers because they are less amenable to rapid experimentation. The corpus needs to be re-indexed every time the expansion technique changes (typically, a costly process); in contrast, manipulations to query representations can happen at retrieval time (and hence are much faster). The success of document expansion has also been mixed; for example, Billerbeck and Zobel (2005) explore both query expansion and document expansion in the same framework and conclude that the former is consistently more effective. A new generation of neural ranking models offer solutions to the vocabulary mismatch problem based on continuous word representations and the ability to learn highly non-linear models of relevance; see recent overviews by Onal et al. (2018) and Mitra and Craswell (2019a). However, due to the size of most corpora and the impracticality of applying inference over every document in response to a query, nearly all implementations today deploy neural networks as re-rankers over initial candidate sets retrieved using standard inverted indexes and a term-based ranking model such as BM25 (Robertson et al., 1994).

[REF4] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: The proposed SparTerm indicates that there is much space for improving the ranking performance of termed-based representations, while still keeping the interpretability and efficiency of BoW methods. Evaluated on MSMARCO dataset, SparTerm significantly outperforms previous sparse models based on the comparable size of PLMs. The top-ranking performance of SparTerm even outperforms Doc2Query-T5, which is based on the pre-trained model of 2x model size and 70x pre-training corpus size. Moreover, we conduct further empirical analysis about how the deep knowledge of PLMs can be transferred to the sparse method, which gives new insights for sparse representation learning. 2 RELATED WORK Our work relates to two research fields: bag-of-words representations and pre-trained language model for text retrieval. 2.1 Bag-of-words Methods Bag-of-words(BoW) methods have played a central role in the firststage retrieval. These methods convert a document or query into a set of single terms, and each term associates a weight to characterize its weight.

[REF5] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: With only the expanded words, SparTerm achieves a definite improvement compared with BM25, especially on Recall. This improvement proves the effectiveness of passage expansion on improving the Recall for retrieval. 5.2 Performance on Document Ranking For the Document Ranking task, we cut down each document into several passages to adapt the max length (256) of the sequence of our model and generate the sparse representation of each passage with our model. We compare our models with two baseline methods: BM25 and HDCT . HDCT is based on the work of DeepCT and focuses on document ranking, which is also a term weighting method. HDCT compares two different ways to combineConference’17, July 2017, Washington, DC, USA Trovato and Tobin, et al.

[REF6] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: INTRODUCTION Text retrieval in response to a natural language query is a core task for information retrieval (IR) systems. Most recent work has adopted a two-stage pipeline to tackle this problem, where an initial set of documents are firstly retrieved from the document collection by a fast retriever, and then further re-ranked by more sophisticated models. For the first-stage retrieval, neural dense representations show great potentials for semantic matching and outperform sparse methods in many NLP tasks, but this is not necessarily true in scenarios that emphasize long document retrieval and exact matching. Moreover, for extremely large (e.g. 10 billion) candidates collection, the dense method has to struggle with the efficiency vs. accuracy tradeoff. Classical term-based sparse representations, also known ∗Both authors contributed equally to this research. †This work is done when Yang Bai is an intern at Huawei Noah’s Ark Lab. Query Can hives be a sign of pregnancy?

[REF7] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: The former two focus on re-weighting while the latter two focus on document expansion: • BM25 is a bag-of-words retrieval models with frequencybased signals to estimate the weights of terms in a text. • DeepCT is a deep contextualized term weighting model which maps the BERT’s representations to term weightings for retrieval. • Doc2query is a document expansion method with Transformer that can expand documents with terms related to the documents’ content. • Doc2query-T5 is a document expansion method which utilizes more powerful T5 language model to generate queries for document expansion. We also evaluate three different settings of SparTerm for evaluation: • SparTerm(literal-only) uses Importance Predictor with the Literal-only Gating which can also be seen as a term weighting model. • SparTerm(expansion-only) uses the Expansion-enhanced Gating for passage expansion without term weighting. We just add the expanded words (weight of each word is 1) to the passages.

[REF8] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: Later proposed methods, such as , , , did not show much advantage over BM25. More recently, Hamed Zamani proposed SRNM to learn a sparse coding in hidden space using weak supervision, which shows good potential for solving the “lexical mismatch” problem. However, the latent unexplainable tokens can not ensure that documents with exact matched terms can be retrieved. 2.2 PLMs for dense text retrieval The pre-trained language models like BERT show new possibilities for text retrieval. Based on dense representations, Lee proposed ORQA with bi-encoder architecture to retrieve candidate passages for question answering using FAISS . However, analysis from concludes that bi-encoders based on dense representation suffer from its capacity limitation in scenarios that emphasize long document retrieval and exact matching.

[REF9] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Document\_expansion\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: as bag-of-words (BoW), such as TF-IDF and BM25 , can efficiently perform literal matching, thus playing a core role in industrial IR systems. However, traditional term-based methods are generally considered to have insufficient representation capacity and inadequate for semantic-level matching. Some attempts have been made to make sparse methods beyond lexical matching while still keeping their advantages. SRNM learns latent sparse representations for the query and document based on dense neural models, in which the “latent” token plays the role of the traditional term during inverted indexing. One challenge about SNRM is that it loses the interpretability of the original terms, which is critical to industrial systems.

........................................................................................................................................................................................................

Title: Learned Sparse Retrieval - Impact score learning

Learned sparse retrieval models have shown significant improvements in retrieval quality while reducing the efficiency gap between neural retrieval and traditional sparse models based on inverted indexes [REF0]. This gap is influenced by vocabulary structures, document expansion techniques, and query expansion strategies, resulting in variations in efficiency among different learned sparse models. To address this, a "guided traversal" approach has been proposed, which combines a learned sparse ranking model with a traditional ranking model [REF0]. This approach utilizes BM25 ranking over a DocT5Query expanded index for index traversal, while employing DeepImpact ranking impacts to compute document scores. Preliminary results demonstrate that the guided traversal approach can match the processing efficiency of traditional sparse models while improving retrieval effectiveness through interpolation of scores from traditional and learned sparse ranking models [REF0].

UniCOIL, a learned sparse retrieval model, has been shown to outperform DeepImpact in terms of retrieval effectiveness [REF1]. Additionally, dense-sparse hybrid models, such as TCTColBERTv2 combined with BM25 and doc2query-T5, have also been explored [REF1]. These findings highlight the ongoing exploration of different configurations and combinations of term weighting models and document expansion techniques in learned sparse retrieval.

Transformer-based ranking methods, such as ColBERT, have been computationally expensive due to the application of transformers to each document during query time [REF2]. In contrast, learned sparse representations aim to approximate the effectiveness of transformer-based methods while retaining the efficiency of traditional bag-of-words rankers like BM25 [REF2]. The main objective of learned sparse representations is to learn the terms under which a document should be indexed (document expansion) and the impact scores that should be stored in the corresponding inverted index postings (learning impacts) [REF2]. By doing so, the resulting ranking function approximates the effectiveness of transformer-based rankers while maintaining the efficiency of inverted-index based methods [REF2].

DeepImpact, a recently proposed learned sparse representation model, addresses the issues of document expansion and term weighting [REF3]. It utilizes document expansion through doc2query-T5 to identify dimensions in the sparse vector that should have non-zero weights [REF3]. The term weighting model of DeepImpact is based on a pairwise loss between relevant and non-relevant texts with respect to a query [REF3]. The model directly predicts term weights, which are then quantized and stored as impact scores in the inverted index postings [REF3]. This approach draws a connection to previous research in information retrieval and emphasizes the importance of learned impacts in the ranking process [REF3].

While learned sparse retrieval approaches have shown effective retrieval, they are often slower than traditional counterparts [REF4]. To address this performance gap, a novel heuristic index traversal mechanism has been proposed, which accelerates DeepImpact retrieval without compromising effectiveness [REF4]. Experimental results demonstrate that this heuristic approach can improve the processing speed of DeepImpact retrieval by a factor of four [REF4].

Learned sparse retrieval techniques offer flexibility in terms of applying expansion and re-weighting to documents and queries [REF5]. For example, DeepCT proposed a regression-based term weighting model without expansion, while DeepImpact combined doc2query-T5 as an expansion model with a term weighting model trained using pairwise loss [REF5]. Exploring different combinations and configurations of expansion and term weighting components can provide insights into the contributions of each component in learned sparse retrieval [REF5].

The advantages and disadvantages of dense and sparse approaches in learned representations for information retrieval are still being explored [REF6]. While dense retrieval techniques aim to maximize inner products between queries and relevant documents using transformer-based encoders, sparse retrieval techniques, such as BM25, rely on exact match ranking models [REF6]. The ongoing research in this area will shed light on the future of learned representations in information retrieval [REF8].

In conclusion, learned sparse retrieval models have shown promising results in improving retrieval quality while maintaining efficiency. The combination of traditional and learned sparse ranking models, along with novel heuristic index traversal mechanisms, has contributed to bridging the efficiency gap between neural and traditional sparse models. Further exploration of different configurations and combinations of expansion and term weighting components will enhance our understanding of the impact of each component in learned sparse retrieval.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF88\_1c68eefcceb042fb79712aed347370d6ce7190c1.pdf Title: Fast Learned Sparse Retrieval with Guided Traversal Chunk of text: 5 CONCLUSIONS Learned sparse models result in substantial retrieval quality improvements while reducing the efficiency gap between neural retrieval and the faster traditional sparse models based on inverted indexes. This gap correlates with vocabulary structures, document expansion techniques, and query expansion strategies, making the several learned sparse models quite different efficiency-wise. In this work, we have proposed a “guided traversal” approach to accelerate index traversal by coupling a learned sparse ranking model with a traditional ranking model. Our proposed approach employs BM25 ranking over a DocT5Query expanded index to lead the index traversal, but uses the DeepImpact ranking impacts to compute document scores. Our preliminary results on top of DeepImpact show that our guided traversal approach is almost able to match the processing efficiency of traditional sparse models, while also improving the retrieval effectiveness of the learned sparse models through interpolation of the scores of the traditional and learned sparse ranking models. In future work, we plan to explore whether our guided traversal heuristic is practical for other learned sparse models, as well as different efficient query processing strategies. Software.

[REF1] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF93\_89d373d61c68465fd49da1257aa959e5abefd155.pdf Title: A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques Chunk of text: To our knowledge, our uniCOIL model, row (2h), represents the state of the art in sparse retrieval using learned impact weights, beating DeepImpact by around two points. The second main block of Table 2 provides a number of comparable dense retrieval results from the literature. The highest score that we are aware of is RocketQA (Qu et al., 2021), whose effectiveness beats all known sparse configurations. Note 3https://github.com/luyug/COIL that ColBERT (Khattab and Zaharia, 2020) uses the more expressive MaxSim operator to compare query and document representations; all other techniques use inner products. The final block of Table 2 presents the results of dense–sparse hybrids. Lin et al. (2021) reported the results of dense–sparse hybrids when TCTColBERTv2, row (3f), is combined with BM25, row (1a), and doc2query–T5, row (1b).

[REF2] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF88\_1c68eefcceb042fb79712aed347370d6ce7190c1.pdf Title: Fast Learned Sparse Retrieval with Guided Traversal Chunk of text: , nearest-neighbor methods for quickly identifying candidate documents in dense retrieval scenarios , and the design of late-interaction transformer-based rankers such as ColBERT . A straightforward application of transformer-based ranking involves applying a transformer at query time to each document that is being re-ranked, leading to significant computational costs. While this can be reduced through approaches such as ColBERT , or by re-ranking only a small set of candidates, the resulting methods are still much more expensive than a simple ranking function that can be directly evaluated over an inverted index. On the other hand, approaches based on learned sparse representations aim to come close to the best transformer-based methods in effectiveness while preserving the efficiency of simple bag-of-words rankers such as BM25 . The main goal in (most) learned sparse representations, as discussed by Lin and Ma , is to learn a set of terms under which a document should be indexed (document expansion), and the impact scores that should be stored in the corresponding inverted index postings (learning impacts), such that the resulting ranking function approximates the effectiveness of a full transformer-based ranker while retaining the efficiency of the fastest inverted-index based methods.

[REF3] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF93\_89d373d61c68465fd49da1257aa959e5abefd155.pdf Title: A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques Chunk of text: These two issues were resolved by the recently proposed DeepImpact model (Mallia et al., 2021), which also belongs in the family of learned sparse representations. DeepImpact brought together two key ideas: the use of document expansion to identify dimensions in the sparse vector that should have non-zero weights and a term weighting model based on a pairwise loss between relevant and nonrelevant texts with respect to a query. Expansion terms were identified by doc2query–T5 (Nogueira and Lin, 2019), a sequence-to-sequence model for document expansion that predicts queries for which a text would be relevant. Since the DeepImpact scoring model directly predicts term weights that are then quantized, it would be more accurate to call these weights learned impacts, since query– document scores are simply the sum of weights of document terms that are found in the query. Calling these impact scores draws an explicit connection to a thread of research in information retrieval dating back two decades (Anh et al., 2001). The recently proposed COIL architecture (Gao et al., 2021a) presents an interesting case for this conceptual framework. Where does it belong?

[REF4] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF88\_1c68eefcceb042fb79712aed347370d6ce7190c1.pdf Title: Fast Learned Sparse Retrieval with Guided Traversal Chunk of text: The main goal in (most) learned sparse representations, as discussed by Lin and Ma , is to learn a set of terms under which a document should be indexed (document expansion), and the impact scores that should be stored in the corresponding inverted index postings (learning impacts), such that the resulting ranking function approximates the effectiveness of a full transformer-based ranker while retaining the efficiency of the fastest inverted-index based methods. Recent work has shown, however, that while effective retrieval is possible with learned sparse approaches, they are often still much slower than their traditional counterparts [20, 22, 27]. In this work, we propose a novel heuristic index traversal mechanism that closes the performance gap between learned and traditional rankers. Experiments over the MSMARCO passage collection demonstrate that our heuristic approach can accelerate DeepImpact retrieval by a factor of four without any measurable loss in effectiveness. 2 BACKGROUND & MOTIVATION We now briefly introduce some background and related work before motivating our proposed approach. arXiv:2204.11314v1

[REF5] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF93\_89d373d61c68465fd49da1257aa959e5abefd155.pdf Title: A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques Chunk of text: Learned sparse retrieval techniques are shown in row group (2). Separating the term weighting component from the expansion component allows us to identify gaps in model configurations that would be interesting to explore. For example, in row (2a), DeepCT proposed a regression-based term weighting model, but performed no expansion. However, the term weighting model can be applied to expanded documents, as in row (2b); to our knowledge, this configuration has not been publicly reported. Similarly, DeepImpact combined doc2query–T5 as an expansion model and a term weighting model trained with pairwise loss. To better understand the contributions of each component, we could run the term weighting model without document expansion, as outlined in row (2c). This ablation experiment was not reported in Mallia et al. (2021), but would be interesting to conduct.

[REF6] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF93\_89d373d61c68465fd49da1257aa959e5abefd155.pdf Title: A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques Chunk of text: Dense Sparse Supervised DPR, ANCE DeepImpact, COIL Unsupervised LSI, LDA BM25, tf–idf Table 1: Our conceptual framework for organizing recent developments in information retrieval. et al., 2021), can be understood as learned dense representations for retrieval. This is formulated as a representational learning problem where the task is to learn (transformer-based) encoders that map queries and documents into dense fixed-width vectors (768 dimensions is typical) in which inner products between queries and relevant documents are maximized, based on supervision signals from a large dataset such as the MS MARCO passage ranking test collection (Bajaj et al., 2018). See Lin et al. (2020) for a survey. Dense retrieval techniques are typically compared against a bag-of-words exact match ranking model such as BM25, which in this context can be understood as unsupervised sparse retrieval. Although it may be unnatural to describe BM25 in this way, it is technically accurate: each document is represented by a sparse vector where each dimension corresponds to a unique term in the vocabulary, and the scoring function assigns a weight to each dimension.

[REF7] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF88\_1c68eefcceb042fb79712aed347370d6ce7190c1.pdf Title: Fast Learned Sparse Retrieval with Guided Traversal Chunk of text: It is important to note that both expansion and re-weighting can be applied to documents (before indexing), to queries (before searching), or any combination thereof. DeepCT [6, 7] is the first example of learned sparse retrieval, exploiting the contextual word representations from BERT to reweight term frequencies for BM25 scoring. The main limitation of DeepCT lies in the fact that it does not address the vocabulary mismatch problem : only terms already appearing in the documents will receive learned weights to improve their relevance signals. DocT5Query addresses the vocabulary mismatch problem by expanding documents offline via the T5 sequence-to-sequence model.

[REF8] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF93\_89d373d61c68465fd49da1257aa959e5abefd155.pdf Title: A Few Brief Notes on DeepImpact, COIL, and a Conceptual Framework for Information Retrieval Techniques Chunk of text: Here, we have only focused on the first aspect. Learned representations for information retrieval are clearly the future, but the advantages and disadvantages of dense vs. sparse approaches along these dimensions are not yet fully understood. It’ll be exciting to see what comes next! 5 Acknowledgments This research was supported in part by the Canada First Research Excellence Fund and the Natural Sciences and Engineering Research Council (NSERC) of Canada. Computational resources were provided by Compute Ontario and Compute Canada.

[REF9] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Impact\_score\_learning/BIBREF91\_4aa1d28944856ebe1950a27f633c6667ead3cbf8.pdf Title: Learning Passage Impacts for Inverted Indexes Chunk of text: We propose DeepImpact, a more effective approach for learning a relevance score contribution for term-document pairs that can also be stored in a classical inverted index. DeepImpact improves impact-score modeling and tackles the vocabulary-mismatch problem between queries and documents. Instead of learning independent term-level scores without taking into account the term co-occurrences in the document, as in DeepCT, or relying on unchanged BM25 scoring, as in DocT5Query, DeepImpact directly optimizes the sum of query term impacts to maximize the score difference between relevant and non-relevant passages for the query. In other words, while DeepCT learns the term frequency component of existing IR models, e.g., BM25, in this work we aim at learning the final term impact jointly across all query terms occurring in a passage. In this way, our proposed model learns richer interaction patterns among the impacts, when compared to training each impact in isolation. To address vocabulary mismatch, DeepImpact leverages DocT5Query to enrich every document with new terms likely to occur in queries for which the document is relevant.

........................................................................................................................................................................................................

Title: Learned Sparse Retrieval - Sparse representation learning

Several approaches have been proposed to enhance traditional sparse Bag-of-Words (BoW) representations using Pre-trained Language Models (PLMs) [REF0]. PLM-based models, such as DeepCT and Doc2query, have been developed to estimate term weights by considering contextualized information and generating document-level term weights [REF0]. These models have shown improvements compared to traditional methods like BM25 [REF0]. However, a key difference between these methods and our work is that DeepCT and Doc2Query train an auxiliary intermediate model to refine sparse representations, while our approach, SparTerm, directly learns sparse representations within the entire vocabulary [REF0].

Sparse representation learning has gained attention due to its ability to combine the advantages of BoW models, such as exact term matching, efficiency of inverted indexes, and interpretability, with the potential for semantic-level matching and reduced vocabulary mismatch [REF1]. By leveraging the deep knowledge of PLMs, SparTerm significantly enhances the ranking performance of term-based representations while maintaining the interpretability and efficiency of BoW methods [REF3]. Moreover, empirical analysis has shown that the transfer of deep knowledge from PLMs to sparse methods provides new insights for sparse representation learning [REF2].

Traditional term-based methods, like TF-IDF and BM25, are widely used in information retrieval systems due to their efficiency in literal matching [REF4]. However, these methods are often considered to have limited representation capacity and inadequate semantic-level matching [REF4]. Sparse Neural Ranking Models (SNRM) have been proposed to address these limitations by learning latent sparse representations based on dense neural models [REF4]. However, SNRM sacrifices the interpretability of original terms, which is crucial for industrial systems [REF4].

In the context of information retrieval, there has been a growing interest in combining the strengths of dense and sparse representations [REF5]. Dense retrieval with approximate nearest neighbors search has shown promising results, but it lacks the ability to explicitly model term matching [REF1]. Sparse representations, on the other hand, offer exact term matching, efficiency, and interpretability [REF1]. Our proposed approach, SparTerm, aims to leverage the benefits of both dense and sparse representations by learning sparse, expansion-aware representations for documents and queries [REF8]. By jointly optimizing ranking and regularization losses, SparTerm achieves state-of-the-art ranking performance among all sparse models [REF9].

In conclusion, learned sparse retrieval methods, such as SparTerm, have emerged as a promising approach to enhance traditional sparse representations in information retrieval. By leveraging the deep knowledge of PLMs and combining the strengths of dense and sparse representations, these methods offer improved ranking performance while maintaining interpretability and efficiency. Further research in this area can provide valuable insights into sparse representation learning and its applications in various information retrieval tasks.

References given to GPT:

[REF0] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: 2.3 PLMs for sparse text retrieval Several PLM-based models have emerged to improve the traditional sparse BoW representations. Dai proposed DeepCT to estimate a term’s weight considering its contextualized information, and this work was later extended to generate document-level term weights . Another work Doc2query tries to “translate” potential queries to expand document content, which also shows a large improvement compared to the traditional BM25 method. The biggest difference between our work and these two methods is that DeepCT and Doc2Query train an auxiliary intermediate model to help refine the sparse representations, while SparTerm is desinged to directly learn sparse representations within the whole vocabulary. 3 SPARSE REPRESENTATION LEARNING This section presents the model architecture of SparTerm and the corresponding training strategy.

[REF1] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF95\_1e8a6de5561f557ff9abf43d538d8d5e9347efa0.pdf Title: SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking Chunk of text: Thus, there have been attempts to substitute standard BOW approaches by learned (neural) rankers. Designing such models poses several challenges regarding efficiency and scalability: therefore there is a need for methods where most of the computation can be done offline and online inference is fast. Dense retrieval with approximate nearest neighbors search has shown impressive results [8, 15, 26], but is still combined with BOW models because of its inability to explicitly model term matching. Hence, there has recently been a growing interest in learning sparse representations for queries and documents [1, 4, 19, 28, 29]. By doing so, models can inherit from the desirable properties of BOW models like exact-match of (possibly latent) terms, efficiency of inverted indexes and interpretability. Additionally, by modeling implicit or explicit (latent, contextualized) expansion mechanisms – similarly to standard expansion models in IR – these models can reduce the vocabulary mismatch. The contributions of this paper are threefold: (1) we build upon SparTerm , and show that a mild tuning of hyperparameters brings improvements that largely outperform the results reported in the original paper; (2) we propose the SParse Lexical AnD Expansion (SPLADE) model, based on a logarithmic activation and sparse regularization.

[REF2] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: We conduct further empirical analysis about how the deep knowledge of PLMs can be transferred to the sparse method, which gives new insights for sparse representation learning. Empirical results show that SAPRT significantly increases the upper limit of sparse retrieval methods. ACKNOWLEDGEMENT We thank Xin Jiang, Xiuqiang He, and Xiao Chen for the helpful discussions. Conference’17, July 2017, Washington, DC, USA Trovato and Tobin, et al.

[REF3] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: The proposed SparTerm indicates that there is much space for improving the ranking performance of termed-based representations, while still keeping the interpretability and efficiency of BoW methods. Evaluated on MSMARCO dataset, SparTerm significantly outperforms previous sparse models based on the comparable size of PLMs. The top-ranking performance of SparTerm even outperforms Doc2Query-T5, which is based on the pre-trained model of 2x model size and 70x pre-training corpus size. Moreover, we conduct further empirical analysis about how the deep knowledge of PLMs can be transferred to the sparse method, which gives new insights for sparse representation learning. 2 RELATED WORK Our work relates to two research fields: bag-of-words representations and pre-trained language model for text retrieval. 2.1 Bag-of-words Methods Bag-of-words(BoW) methods have played a central role in the firststage retrieval. These methods convert a document or query into a set of single terms, and each term associates a weight to characterize its weight.

[REF4] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: as bag-of-words (BoW), such as TF-IDF and BM25 , can efficiently perform literal matching, thus playing a core role in industrial IR systems. However, traditional term-based methods are generally considered to have insufficient representation capacity and inadequate for semantic-level matching. Some attempts have been made to make sparse methods beyond lexical matching while still keeping their advantages. SRNM learns latent sparse representations for the query and document based on dense neural models, in which the “latent” token plays the role of the traditional term during inverted indexing. One challenge about SNRM is that it loses the interpretability of the original terms, which is critical to industrial systems.

[REF5] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: INTRODUCTION Text retrieval in response to a natural language query is a core task for information retrieval (IR) systems. Most recent work has adopted a two-stage pipeline to tackle this problem, where an initial set of documents are firstly retrieved from the document collection by a fast retriever, and then further re-ranked by more sophisticated models. For the first-stage retrieval, neural dense representations show great potentials for semantic matching and outperform sparse methods in many NLP tasks, but this is not necessarily true in scenarios that emphasize long document retrieval and exact matching. Moreover, for extremely large (e.g. 10 billion) candidates collection, the dense method has to struggle with the efficiency vs. accuracy tradeoff. Classical term-based sparse representations, also known ∗Both authors contributed equally to this research. †This work is done when Yang Bai is an intern at Huawei Noah’s Ark Lab. Query Can hives be a sign of pregnancy?

[REF6] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF96\_9f753f67da834e59f9a5c8cdf9a88ee84c496b2d.pdf Title: Minimizing FLOPS to Learn Efficient Sparse Representations Chunk of text: λFe(fθ, D) | {z } Le(θ) . (4) Sparse retrieval and re-ranking. During inference, the sparse vector of a query image is first obtained from the learned model and the nearest neighbour is searched in a database of sparse vectors forming a sparse matrix. An efficient algorithm to compute the dot product of the sparse query vector with the sparse matrix is presented in Figure 1. This consists of first building a list of the non-zero values and their positions in each column. As motivated in Section 3, given a sparse query vector, it is sufficient to only iterate through the non-zero values and the corresponding columns. Next, a filtering step is performed keeping only scores greater than a specified threshold.

[REF7] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: Later proposed methods, such as , , , did not show much advantage over BM25. More recently, Hamed Zamani proposed SRNM to learn a sparse coding in hidden space using weak supervision, which shows good potential for solving the “lexical mismatch” problem. However, the latent unexplainable tokens can not ensure that documents with exact matched terms can be retrieved. 2.2 PLMs for dense text retrieval The pre-trained language models like BERT show new possibilities for text retrieval. Based on dense representations, Lee proposed ORQA with bi-encoder architecture to retrieve candidate passages for question answering using FAISS . However, analysis from concludes that bi-encoders based on dense representation suffer from its capacity limitation in scenarios that emphasize long document retrieval and exact matching.

[REF8] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF95\_1e8a6de5561f557ff9abf43d538d8d5e9347efa0.pdf Title: SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking Chunk of text: We propose to combine the best of both worlds for end-to-end training of sparse, expansion-aware representations of documents and queries. Thus, we discard the binary gating in SparTerm, and instead learn our log-saturated model (Eq. 4) by jointly optimizing ranking and regularization losses: L = L𝑟𝑎𝑛𝑘−𝐼𝐵𝑁 + 𝜆𝑞L 𝑞 reg + 𝜆𝑑L 𝑑 reg (6) where Lreg is a sparse regularization (ℓ1 or ℓFLOPS). We use two distinct regularization weights (𝜆𝑑 and 𝜆𝑞) for queries and documents – allowing to put more pressure on the sparsity for queries, which is critical for fast retrieval. 4 EXPERIMENTAL SETTING AND RESULTS We trained and evaluated our models on the MS MARCO passage ranking dataset1 in the full ranking setting.

[REF9] - paperID: ./papers\_pdf/paper\_section/Learned\_Sparse\_Retrieval-Sparse\_representation\_learning/BIBREF87\_57a07372e2a620d6ae920f74877eee5f61753a96.pdf Title: SparTerm: Learning Term-based Sparse Representation for Fast Text Retrieval Chunk of text: Figure 4: The Top 5 contributing words to the expanded words of the second case in Figure 3. The X-axis are the words in the passage and Y-axis represents logit. 6 CONCLUSION In this work, we propose SparTerm to directly learn term-based sparse representation in the full vocabulary space. SparTerm learns a function to map the frequency-based and BoW representation to a sparse term importance distribution in the whole vocabulary space, which involves both term-weighting and expansion in the same framework. Experiments conducted on MSMARCO dataset show that SparTerm significantly outperforms previous sparse models based on the comparable size of PLMs, achieving state-of-the-art ranking performance among all sparse models. We conduct further empirical analysis about how the deep knowledge of PLMs can be transferred to the sparse method, which gives new insights for sparse representation learning. Empirical results show that SAPRT significantly increases the upper limit of sparse retrieval methods.

........................................................................................................................................................................................................