1 Introduction

In this report we compare the traditional methods for Information Retrieval (IR) with newer neural approaches to IR tasks. Our goal would be to contrast traditional IR with neural IR and see whether it can be reasonably expected that neural IR has access to some novel signals or patterns that traditional IR lacks. We use two representative neural models i.e. Kernel Neural Retrieval Model (KNRM) [Xio+17] and Deep Relevance Matching model (DRMM) [Guo+16].

In sections 2–4 , we briefly present the two neural models and discuss their similarities and differences. Sections 5,6 expand on this information and probe the two models through axioms presented in [FTZ04]. Finally, we compare the two neural models with traditional IR in sec 7,8 and present our conclusions.

2 Deep Relevance Matching model

DRMM is an interaction focused model. It builds a pairwise interaction matrix of query and document terms by computing cosine-similarity (interaction) between the embeddings of each pair of query and document terms.

This intermediate matrix representation depends on the lengths of query and document and is thus not suitable as neural net input. Thus, it is then transformed into a fixed-length histogram representation based on matches at multiple similarity levels. The similarity space [-1,1) is divided into fixed length bins and a special bin [1,1] is assigned to exact matches. This histogram is passed to a feed forward network which produces a matching score for each query term.

The individual query term's matching scores are combined to produce a single value by weighting the individual scores with query term's IDF and summing them.

3 Kernel Neural Retrieval Model

KNRM is also an interaction focused model and similarly to DRMM it first builds an interaction matrix. It is then converted into a fixed size representation by applying K kernels to each query term's row.

The kernel function used is RBF. It may also be called the local response function because of its high response to values near kernel level (μ) and vanishing rapidly as value diverges from μ. K applications of this kernel at different μ levels results in a K-dimensional representation for each query term's interaction row.

For a query of length n, these n K-vectors are combined through a log-sum to produce a single K-dim feature vector for neural net input. This feature vector is used by single layer network (a linear classifier) to produce the final matching score.

4 Comparison between DRMM and KNRM

Comparing both methods, we observe that there are many similarities. Both methods treat the query and document terms as bag-of-words (BoW) and start by building a local interaction matrix from word embeddings. At this point, both models lose access to term identity information and only deal with pairwise interaction statistics. Then they convert it into a fixed length histogram representation based on multi-level similarity matches through a pooling method. This representation is then used by a feed forward network as query-document ranking features to produce a ranking score for each query term.

The key difference in both models is that DRMM uses fixed embeddings, whereas in KNRM the word2vec initialized embeddings are also learned during training. These trainable embeddings represent the main learning capacity of KNRM with 49 million parameters, compared to only 155 parameters in DRMM. The training of embeddings in KNRM is made possible by the use of a differentiable soft-histogram.

As discussed in [Xio+17] trainable embeddings allow KNRM to capture domain specific semantic similarities. Through this KNRM is able to decouple several words and bring some other words closer together which were not considered related by word2vec. This also lends support to the idea of similarity vs. relevance matching presented in [Guo+16] that in IR tasks the notion of being "similar" is different from NLP applications. Thus, the performance difference between DRMM and KNRM is not because of KNRM learning to identify some novel signals, but is a result of fine tuned embeddings.

5 Axiomatic analysis of DRMM

Term Frequency Constraints

TFC1: It can be satisfied if the network has learned to recognize the importance of exact match signals. Ideally, the network would assign a positive weight to input node corresponding to exact matching bin i.e. [1,1]. Thus whenever a document d_1 has more occurrences of exact query term matches than d_2, its matching histogram will have a greater number in corresponding component and it will get higher score.

TFC2 is ensured as a result of using logarithmic scaling of counts in the matching histogram bins. It ensures diminishing returns for increasing number of matches.

Term Discrimination Constraints

DRMM explicitly uses the query term's IDF to weight the score produced by network. Hence, the network has access to discrimination statistics and can be expected to learn to give more importance i.e. produce a higher score for matches of rare terms.

Length Normalization Constraints

LNC1: It is likely to be violated because it does not take into account that the added term can have soft matches with query terms. Consider two documents d_1 and d_2, both having same number of exact matches. Now if a soft match term with high similarity is added to d_2 (e.g "lunch" is added to the document under query "order food") it is possible that under the learned network parameters it ranks higher than d_1. We also note that this is the desired behavior under semantic matching axioms in [FZ06].

When the added term is unrelated, the effect of addition depends on precisely which similarity bin it was assigned and what weights the network has to those input nodes. We can expect that the network will learn to ignore spurious matches, otherwise it will be biased towards long documents and the scoring function will be dominated by contamination from spurious matches.

LNC2: In case of LNC2, we see that concatenating a document with itself k times will result in a k-fold increase in the histogram counts (including exact matches). This, combined with TFC1 ensures that the score will increase. However, the discounting effect of logarithmic scaling also noted in TFC2 makes sure that this does not lead to a blowup in relevance score.

TF-Length Constraint

Two documents d_1 and d_2 which have different lengths and where d_1 was constructed by adding more query terms to d_2 will have their matching histograms differing only in the exact matching bin. Thus, as has been discussed before under the right network parameters d_1 will rank higher than d_2. The longer length of d_1 will not in itself penalize it.

6 Axiomatic analysis of KNRM

Term Frequency Constraints

TFC1: KNRM uses a very small kernel width of 10^{-3} for the exact match case i.e $\mu = 1$, and by so doing it preserves the exact matching signal uncontaminated throughout kernel pooling. For the case of a query containing only one term we see that the feature vector produced by kernel pooling will have strong contribution from exact matching bin. Thus using the same argument as in DRMM, we expect the model to give precedence to this bin and produce a higher score.

TFC2: As in DRMM, logarithmic scaling feature vector ensures diminishing returns for increasing number of matches.

Term Discrimination Constraints

KNRM does not appear to have access to any term importance statistics in its ranking function. The interaction pooling results in loss of term identify and no external informa-

tion about term discrimination is provided. Thus we expect the model to treat all term matches as the same.

Length Normalization Constraints

LNC1: Using the same argument as in DRMM, we expect LCN1 to be violated in favor of assigning a higher score to semantically matching document.

LNC2: Similar to DRMM, we can see that concatenating a document with itself k times will result in a k-fold increase in the exact matching kernel. This should lead to a higher score. Log scaling of the pooled feature vector has a regularizing effect on input and prevents unbounded increase.

TF-Length Constraint

As in DRMM, the vectors produced by kernel pooling will differ only in the exact matching component. When the pooled K-dimension vectors are log-summed the higher value will be passed to the network. Using the same argument as before, we can expect that KNRM's learning to rank layer will produce a higher score for document having more term matches.

7 Comparison of neural models with traditional IR

Traditional IR models like BM25 use three important signals in different combinations to produce a relevance score: term frequency, term discrimination, document lenght. First we try to answer the question: At the very least, do neural models have access to the three important IR signals that traditional models including BM25 use?

Term frequency

It can be clearly seen that in both DRMM and KNRM the representations are a monotonic function of matches between query term and document terms. In both models one of the input vector components represents the exact match term frequency. In DRMM for example, the matching histogram mapping explicitly distinguishes between the exact and soft match and keeps the counts of exact match in a separate bin. Thus the neural nets parameters W can, in principle, learn to give more importance to contributions originating from this input node.

In KNRM as well, we observe that model specifically take an exact match into account by setting the width of kernel to $\sigma = 1e - 3$, hence essentially localizing the response to a single point.

Term discrimination

DRRM explicitly uses query term's IDF to weight the scores produced by the network. Therefore, IDF has a direct bearing on the final score produced by DRMM.

Interestingly, KNRM does not make an explicit use of IDF or any IDF like measure. This leads to an interesting question: What (if anything) is performing the role of IDF in KNRM?. The most we can say is that it could not have been encoded in learned embeddings, because the models loses term distinction in interaction matrix.

Document length

DRMM encodes the information about document length in the constructed matching histogram mapping. The sum of all histogram entries gives the document length. The best performing DRMM variant uses *log* to normalize the counts. Thus instead of explicitly passing the document length information to network, it utilizes a transformed multiplicative document count, where the measure of length is given not by sum of term counts, but by multiplication of all counts.

In KNRM, we see that it does not have explicit access to document length information. However, the model can be made robust to variations in different document size by choosing the kernel width σ so that the kernel response is well localized. By doing so, most irrelevant interactions will be reduced to zero in the kernel pooling step.

8 Do neural IR models learn novel new functions?

In this section we try to identify whether neural models learn to identify any new signals or patterns that can help in IR tasks.

Semantic soft matching

Based on the results presented in [Guo+16] and [Xio+17], we can say that semantic similarity matching (soft matching) is a desirable property for models to have and an important signal of relevance. This is also supported by other methods which make use of it [Hu+14] [She+15] [KR15] [Zuc+15]. Neural models assume access to pre-trained word-embeddings for soft-matching. In absence of this, the models are essentially reduced to exact matching models. KNRM further tunes the embeddings so they better reflect the underlying semantic structure of task at hand and the notion of similarity is adapted to the domain. Models which combine both exact and soft matching tend to perform better than those lacking in either one of these [MC18].

However, we also note that integrating semantic matching in IR is not specific to neural models. Semantic matching in traditional approaches has been shown in [FZ06] along with a set of axioms to guide the choice of similarity function and its effect on the scores.

Phrase and sentence matches

As noted earlier, DRMM and KNRM are both BoW models and thus do not have access to higher level structural or semantic information. Some other models inspired by CNNs ability to capture positional regularities in image patches have been applied to the sentence

matching problem with the goal of learning phrase matching and hierarchical meanings [Pan+17] [Hu+14]. In these models the knowledge about language and sentence structure is baked in to the model design and they can learn to recognize phrasal structures by looking at several terms at once. This goes beyond capability of traditional BoW models. Using this phrase match as building block these models at higher levels of granularity learn to identify similar sentences.

Use of behavioural data

Neural IR model can combine query and document statistics with user behaviour and context data such as click-through rate (CTR), dwell time, bounce-rate, document interaction measures etc. to provide relevance judgement. However, in such models the distinction between behavioural models and IR models gets blurred and it would not be fair to compare them with models which only take into account query-document statistics.

Efficiency and bootstrapping

Supervised neural models require well defined relevance judgements beforehand in the form of labelled training data which can be very challenging to acquire. These networks also may have millions of parameters (as in KNRM) compared to very few parameters in BM25.

Additionally, neural models cannot be used to explore a new previously unseen corpus as without labelled training they have no way of identifying relevant documents in it (this can be alleviated to some degree by use of transfer learning, but we have not come across this in the literature considered for this report). Whereas, BM25 can function after some simple statistics (e.g TF, IDF, document length) have been computed without the need for precise relevance judgement. Thus we remark that traditional models can be used to seed the training data where top (resp. bottom) documents returned by BM25 are given to network as positive (resp. negative) examples under a query.

9 Conclusion

In conclusion we do not observe any significant differences from traditional approaches in DRMM and KNRM with the exception of soft matching.

References

[FTZ04] H. Fang, T. Tao, and C. Zhai. "A Formal Study of Information Retrieval Heuristics". In: *SIGIR* (2004), pp. 49–56.

[FZ06] H. Fang and C. Zhai. "Semantic term matching in axiomatic approaches to information retrieval". In: *SIGIR* (2006), pp. 115–122.

[Hu+14] B. Hu et al. "Convolutional Neural Network Architectures for Matching Natural Language Sentences". In: *NIPS* (2014), pp. 2042–2050.

[KR15] T. Kenter and M. de Rijke. "Short text similarity with word embeddings". In: *CIKM* (2015), pp. 1411–1420.

[She+15] Y. Shen et al. "Learning semantic representations using convolutional neural networks for web search". In: *CIKM* (2015), pp. 1411–1420.

[Zuc+15] G. Zuccon et al. "Integrating and Evaluating Neural Word Embeddings in Information Retrieval". In: *Proceedings of the 20th Australasian Document Computing Symposium (ADCS)* (2015), p. 12.

[Guo+16] J. Guo et al. "A deep relevance matching model for ad-hoc retrieval". In: *Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM)* (2016), pp. 55–64.

[Pan+17] L. Pang et al. "Text matching as Image Recognition". In: *Proceedings of the 13th AAAI Conference on Artificial Intelligence (AAAI)* (2017), pp. 2793–2799.

[Xio+17] C. Xiong et al. "End-to-End Neural Ad-hoc Ranking with Kernel Pooling". In: *Proceedings of the 40th International ACM SIGIR Conference on Research Development in Information Retrieval* (2017).

[MC18] Bhaskar Mitra and Nick Craswell. *An Introduction to Neural Information Retrieval*. 2018. URL: `https://www.microsoft.com/en-us/research/uploads/prod/2017/06/fntir2018-neuralir-mitra.pdf`. (accessed: 22.05.2019).