Modelling document-query interaction in a hierarchical neural model for IR

Johan Chagnon, Diana Nicoleta Popa, Yagmur Gizem Cinar, Éric Gaussier

Modelling document-query interaction in a hierarchical neural model for IR

Johan Chagnon[*] — Diana Nicoleta Popa[*] — Yagmur Gizem Cinar[*] — Eric Gaussier[*]

[*] *Universite Grenoble Alpes*

RÉSUMÉ. *Les techniques récentes dans le domaine de la recherche d'information adoptent une approche reposant sur la représentation ou l'interaction, en fonction de ce qui est le plus adapté à l'exercice. Dans cet article, nous présentons une approche hiérarchique pour la représentation des documents permettant de modéliser l'interaction document-requête à différents niveaux de granularité. Le modèle présenté sépare chaque document en un ensemble de blocs qui sont appariés à la requête donnée à l'aide de modules d'attention et de couche de pooling et de projection. Nous avons évalué notre modèle sur le jeu de données de LETOR 4.0 MQ2007. L'approche montre des résultats préliminaires prometteurs, bien qu'une exploration plus approfondie des choix de modélisation pourrait apporter des gains supplémentaires.*

ABSTRACT. *Recent deep approaches to information retrieval are either representation-oriented or interaction-oriented, depending on how they view the modelling of document and query representations and their interactions. We explore a hierarchical approach to document encoding that enables modelling the query-document interaction at different levels of granularity. The proposed model splits the input documents into blocks that are individually matched to a given query through a series of self-attention modules, along with pooling and projection layers. We test our method on the LETOR 4.0 MQ2007 standard IR collection. The approach shows promising preliminary results, albeit a more in-depth exploration of the modelling choices could provide further gains.*

MOTS-CLÉS : *encodage hierachique de documents, self-attention, interaction document-requête*

KEYWORDS: *hierarchical document encoding model, self-attention, query-document interaction.*

1. Introduction

Several advanced deep learning models have been proposed recently to encode longer sequences of text such as paragraphs and documents (Yang *et al.*, 2016 ; Chang *et al.*, 2019 ; Zhang *et al.*, 2019 ; Beltagy *et al.*, 2020). Such methods are often characterized by complex architectures with a large number of parameters that aim at properly encoding long textual units. Moreover, most of these methods have been mainly evaluated on tasks based on document level similarity assessment, entailment recognition, question-answering and summarization. At the same time, much work has been done recently on modelling the interaction between queries and documents within the information retrieval (IR) space, but with less emphasis within the same models on the depth and complexity of the used architectures for representation learning. Some exceptions exist though, such as DeepRank (Pang *et al.*, 2017) and PARADE (Li *et al.*, 2020), which leverage either the Transformer architecture (Vaswani *et al.*, 2017) or a combination of deep convolutional and recurrent neural networks.

In the current work we propose a model for learning query-informed document representations for IR, thus targeting both representation learning and interaction modelling within the same model. Our proposal is lighter in terms of parameters as compared to related work that focuses on both aspects. The self-attention mechanism (Vaswani *et al.*, 2017), very popular in recent deep models, is used to model the interaction between queries and documents at different granularity levels throughout the construction of their representations. Additionally, pooling and projection layers enable feature selection and transformations at the same granularity levels. Finally, we choose to directly optimise losses that represent differentiable approximations of standard IR metrics (in this case P@K and NDCG@K). Preliminary results obtained are promising, but more effort needs to be put into design choices in order to reach state-of-the-art scores.

The remainder of the document is organised as follows: Section 2 presents the related work, while the proposed base model is detailed in Section 3, along with alternatives for interaction modelling presented in Subsection 3.3. The experimental setup and the data used for evaluation are presented in Section 4, while the results are detailed in Section 5. We draw conclusions and showcase potential future directions in Section 6.

2. Related work

Deep models for information retrieval can be divided into two categories depending on how they model the interactions of documents and queries and their representations. The representation-oriented models (Huang *et al.*, 2013 ; Shen *et al.*, 2014 ; Severyn et Moschitti, 2015) first focus on accurately modelling the documents and queries in isolation, often using siamese architectures, followed by the computation of the relevance of each document for a given query. The interaction-oriented models (Guo *et al.*, 2016 ; Xiong *et al.*, 2017 ; Fan *et al.*, 2018 ; Li *et al.*, 2020) aim at extracting features regarding the document and query and their interaction at earlier stages of their representation building, thus going beyond a semantic matching-based approach to IR. Our proposal fits into the second category as we aim at modelling the document-query interaction and their representations jointly. We consider that the complexity addition given by the interaction-based approach as opposed to the representation-oriented ones is compensated by the higher flexibility and access to more granular and informed modelling such an approach entails.

Among the representation-oriented models, DSSM (Huang *et al.*, 2013) maps documents and queries into a common space using a siamese network and computes the relevance between them through the means of cosine similarity, while (Severyn et Moschitti, 2015) jointly encode the query and documents along with a tunable similarity matrix followed by a multi-layer perceptron (MLP). CDSSM (Shen *et al.*, 2014) uses word n-grams that are further fed to a CNN architecture aimed at preserving word order and capturing context information, with ARC-I and ARC-II (Hu *et al.*, 2014) also using a convolutional-based approach. While ARC-I has the drawback of deferring the interaction modelling until later stages, once the individual representations are formed, ARC-II aims at circumventing this aspect by allowing an earlier interaction through the concatenation of representations obtained through shared convolutions.

(Guo *et al.*, 2016) propose DRMM, a matching model that learns the interaction between matching features and takes into account exact matching signals, query-term relevance and diverse matching requirements. (Xiong *et al.*, 2017) use soft matches from query-document interactions at word level, while (Mitra *et al.*, 2017) propose using two neural network architectures trained jointly: a local model for exact matching and a distributed model for semantic matching. (Pang *et al.*, 2017) propose DeepRank, a method that simulates the human judgement process, using a combination of convolutional and recurrent modules on top of query-centric contexts. (Fan *et al.*, 2018) explicitly model relevance signals at different granularity levels (document level and passage level), by providing a local matching component and a global decision component. The local matching uses a bidirectional GRU over query-passage matching tensors, while the global component accumulates the local signals using a recurrent architecture. (Li *et al.*, 2019) use a similar local matching component, but in a model leveraging human reading patterns during relevance judgements, that incorporates explicit heuristics learned by reinforcement learning. Our proposal falls into the category of interaction-oriented models, while at the same time differing from the abovementioned models through the choice of architectural design: we choose to employ a hierarchical architecture, much lighter in terms of parameters than recent models using Transformers, while at the same time benefiting from the expressiveness of the self-attention mechanism. Our proposal also allows modelling weaker relevance signals that are not exact matches of query words, unlike related work such as DeepRank.

3. Hierarchical document-query model

Documents are composed of sub-units of text such as words, sentences and paragraphs. The local relevance of a part of a document to a query contributes to and determines the (global) relevance of this document to the query. In order to capture such local and global relevance aspects, we adopt a hierarchical document-query interaction representation. The architecture is composed of a local self-attention layer, a local pooling operation and a local projection, followed by a document-level global self-attention layer, global pooling and projection operations. Figure 1 presents the base model, while alternative design choices are explored further in Subsection 3.3.

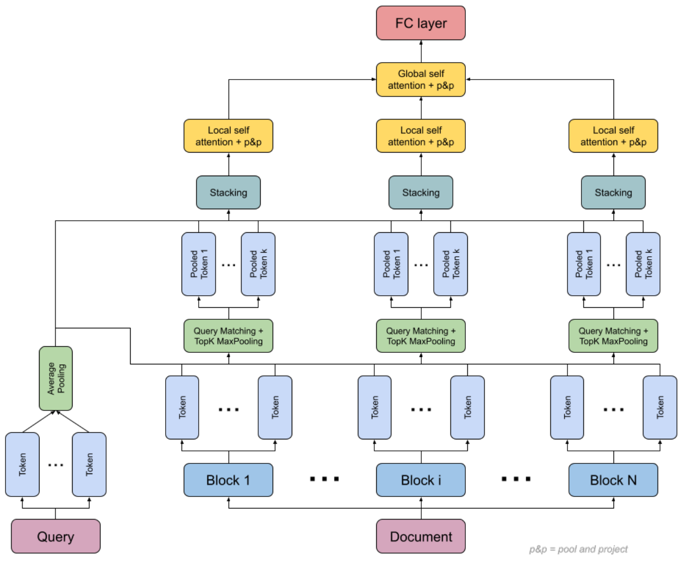

Figure 1. *Architecture of the base model*

3.1. *Preliminaries: document segmentation*

To capture local relevance signals in a document with respect to a query we aim at enabling the access to sub-units of that document and modelling their individual interaction with the given query. To do so, we first divide the document into blocks of maximum N_{tokens} tokens: (i) append consecutive full sentences to a block as long as the maximum number of tokens it contains remains smaller than N_{tokens} (hyper-parameter), (ii) create new blocks by repeating (i) until the end of the document is reached. Whenever the number of tokens of a sentence exceeds N_{tokens}, the sentence is divided across several blocks. Padding is also done at the block level whenever necessary. After this, one obtains a set of tokens that constitute the set of blocks. In the Query Matching and Top K MaxPooling module, the top k most relevant tokens of each block are then pooled using cosine similarity with respect to the query. This allows a smaller and fixed representation space of the block while keeping most of the relevance signals. When the number of tokens to pool is bigger than the number of tokens of the block, the tokens are repeated until k is reached. A block is first represented by stacking the k n-dimensional pooled tokens representations $b_i^1 \in \mathbb{R}^{n \times k}$.

3.2. *Document-query matching*

In order to enable interactions between the query and the different parts of the document (namely the different blocks in the current setup), the representation of each block is enhanced by stacking the query representation on top of it. To represent the query, we use average pooling across all the tokens it contains. A block representation thus becomes $b_i \in \mathbb{R}^{n \times (k+1)}$.

The document-query matching phase is performed by leveraging a local self-attention layer. Using self-attention enables contextualizing the embeddings of the pooled tokens of the block, while at the same time allowing the query representation to influence their representations. We use here the now standard self-attention equation of (Vaswani *et al.*, 2017).

After applying the local self-attention layer, a block is still represented by a two-dimensional matrix $b_i \in \mathbb{R}^{n \times (k+1)}$. 1D-Maxpooling is further applied in order to obtain a single feature vector per block $b_i \in \mathbb{R}^n$, allowing the most pertinent features to be considered. The result is projected into a new space by applying a linear transformation using $W_{local} \in \mathbb{R}^{n \times q}$, thus obtaining a new block representation $b_i \in \mathbb{R}^q$. Further, a global self-attention layer is considered to capture long-distance dependencies that span across the whole document. The global self-attention acts upon and updates the different block representations, forcing them to be contextualized with respect to each-other. The result of this layer constitutes the document-query representation $dq \in \mathbb{R}^{q \times N_{bl}}$, where N_{bl} represents the number of blocks in the document. After applying 1D-Maxpooling, the updated document-query representation $dq \in \mathbb{R}^q$ is obtained. This is further transformed through a global projection matrix $W_{global} \in \mathbb{R}^{q \times s}$, thus obtaining $dq \in \mathbb{R}^s$. The result is finally fed into a fully connected layer to estimate the relevance value of the document-query pair.

3.3. *Model variations*

We have additionally considered different design choices within the above model. We considered using a different pooling module before the self attention, to represent each block in a lower dimensional space. After the block segmentation, each block b_i is divided into a fixed number of sub-blocks N_{sb}. Average pooling is performed over the embeddings of the tokens within each sub-block sb_j such that one single n-dimensional vector is obtained as its representation $sb_j \in \mathbb{R}^n$. A block is then first represented by the stacking of the sub-block representations $b_i^1 \in \mathbb{R}^{n \times N_{sb}}$, to which we add the query representation. This earlier version of the model has been shown to yield a lower performance, probably due to the trade-off between capturing all relevance signals of a block and averaging it out.

An alternative to the 1D-Maxpooling consists in using the Topk-Maxpooling module presented previously: instead of only selecting the most salient feature, select the top k ones, where saliency is measured in terms of cosine similarity with the query (excluding the transformed query token). One option is to remove the 1D-Maxpooling layer, stack together the output of all local self-attention modules and then pool the top k_{local} most relevant features with respect to the query. Another possibility is to keep a fixed number of features per blocks before stacking them together. A combination of both possibilities has also been explored, but without improvements.

Additional feature vectors can also be easily integrated in the current architecture. One example is that of the the learning-to-rank feature vector present in the MQ2007 dataset which contains information on the document, the query and the relation between them. It can be concatenated to the computed feature representations obtained by the proposed model either at earlier stages or at later stages, following the global projection layer.

Replacing the self-attention layers with Transformer (Vaswani *et al.*, 2017) layers is also an option. However, in practical terms, this enhancement did not yield an improved performance throughout the experiments we carried. Nevertheless, maintaining the self-attention layer instead of the full Transformer layer provides a gain in speed and simplicity. Finally, we also considered using the individual word embeddings for the query terms instead of their averaged representation to match fine-grained signals. In practice though, this did not yield an increased performance.

4. Experimental setup

Dataset Similarly to related work, we evaluate our method on the MQ2007 data, a subpart of the LETOR 4.0 dataset (Qin *et al.*, 2010) which contains data sampled from the GoV2 web page collection ($\sim$ 25M pages). The set of queries used can be found at (NIST, 2008), while the set of qrels, linking a query-document pair to its relevance label, is taken from (Microsoft, 2009). Within MQ2007, there are a total of 1,692 queries and 65,323 labeled documents. There are 3 relevance labels considered $\{0,1,2\}$, with the property that the larger the relevance label is, the more relevant the query-document pair.

Preprocessing All documents were extracted from the HTML pages using BeautifulSoup4[1]. All words were converted to small cases and very large documents were truncated to a limit of around 2500 tokens. This concerned 21.5% of documents and did not have a significant influence on the final performance scores. One reason behind is that large documents are full of extraction artifacts.

1. https://pypi.org/project/beautifulsoup4/

Implementation details Throughout our experiments, words are represented using the 300-dimensional GloVe (Pennington *et al.*, 2014) embeddings, pre-trained on the Wikipedia 2014 and Gigaword 5 datasets. BERT (Devlin *et al.*, 2018) embeddings have also been used as an alternative, but without substantial improvement in the results. We follow the standard train/validation/test split of MQ2007 as given per LETOR 4.0 (Microsoft, 2009) which uses approximately 60% of data for training, 20% for the validation and 20% for testing.

Two separate grid searches have been performed on the validation set to determine the best parameter values. The first grid search aimed at finding the best combination of $N_{tokens} \in \{50, 100, 200, 400\}$ and the number of pooled tokens per block $k \in \{4, 8, 16, 32, 64, 128\}$. The best results were obtained when having a small amount of blocks (i.e. a big value for N_{tokens}) and a moderate to big amount of pooled tokens. For the rest of the experiments the following values were used: $N_{tokens} = 200$ and $k = 16$ as they appeared to be the best trade-off between complexity and performance. Having a moderate number of pooled tokens was expected to perform well as there is a compromise between how much we want the query to influence the document representations within the local self attention layer, and how much emphasis we want to put on a precise representation of the block. The second grid search aimed at optimizing the size of the global self attention input and output values. Values considered for both were in the set $\{64, 128, 256, 512\}$. We found out that the model performed better when the global output and input dimensions were larger ($global\_input\_dim = 512$, $global\_output\_dim = 512$), although having a large output space seems more impactful.

Lastly, the learning rate was set to 0.001 and the output of the fully connected layer is of dimension one, as we aim to give a numerical relevance value to the documents. The total number of tunable parameters of the proposed model is 1,830,129. This can be changed as a lower global output dimension can be considered at the cost of very little performance difference according to preliminary tests.

Loss The network is trained by using a differentiable approximation of the IR metrics (*e.g.* P@K, NDCG@K) (Thonet *et al.*, n.d.), which is formulated using an iterative soft rank indicator. The precision P@K loss and normalized discounted cumulative gain NDCG@K loss are defined as $\mathcal{L}_{\text{P@K}} = 1 - \widehat{\text{P@K}}$ and $\mathcal{L}_{\text{NDCG@K}} = 1 - \widehat{\text{NDCG@K}}$, respectively. $\widehat{\text{P@K}}$ and $\widehat{\text{NDCG@K}}$ are the differentiable approximations of P@K and NDCG@K IR metrics. Similarly to related work, one could employ a pairwise loss instead of the listwise approach. We leave the exploration of such an approach to future work.

5. Results

Table 1 shows the results obtained using the model presented in Figure 1 and the best parameters as selected on the validation set. These results have been computed on the standard split of MQ2007 and represent the average of scores over the 5 folds.

Model	nDCG1	nDCG5	nDCG10	nDCG	P1	P5	P10
DSSM	0.290	0.335	0.371	-	0.345	0.359	0.352
CDSSM	0.288	0.297	0.325	-	0.333	0.301	0.291
DRMM	0.380	0.408	0.440	-	0.450	0.417	0.388
Arc-II	0.317	0.354	0.390	-	0.379	0.377	0.366
DeepRank-2DGRU	0.439	0.447	0.473	-	0.513	0.443	0.405
DeepRank-CNN	0.441	0.457	0.482	-	0.508	0.452	0.412
Hierarchical Attention	0.320	0.369	0.427	0.553	0.365	0.360	0.343

Tableau 1. *Performance comparison of the proposed model and related work on the test set. Results of related work are taken from (Pang* et al.*, 2017).*

As one can note, representation-focused models such as DSSM and CDSSM perform worse than our proposed model. DRMM and Arc-II, which are interaction-focused, show comparable results whereas both versions of DeepRank have better performances. The design of our hierarchical attention model is somewhat close to DeepRank: both models have a local matching phase which is followed by an aggregation layer. One main difference between the architectures, though, is that instead of using block division as in the current proposal, DeepRank uses a query-centric context[2] That means that in DeepRank blocks are not a partition of the document but instead represent areas of interest, which seems to create more accurate embeddings. Thus DeepRank does not focus only on the most relevant tokens but also takes benefits from their context. The downside however is that their model has to match exactly a query word in order to create a context. One asset of our model is that it does not require exact matching and thus is able to treat synonyms or close words appropriately.

6. Conclusion

In this paper, we proposed a hierarchical document-query architecture for IR that relies on locally collected relevance signals that are aggregated and combined at a global level. Preliminary results on the MQ2007 dataset show that our model has comparable performance to related work. However, while enhancements have been proposed for the model, current design choices do not provide state-of-the-art results. The main reason behind the score differences with respect to the state-of-the-art method DeepRank should be that the use of query-centric contexts allows a precise and exhaustive embedding of the words of interest and their relevance, which eases the learning. However our architecture should enable capturing relevant tokens of different degrees of relevance, not just exact matches. One could imagine that instead of only pooling the top k relevant tokens of each block, we capture part of their context as well, which would represent an interesting trade-off between architectures.

2. DeepRank defines a query-centric context as a window centered on a query term occurrence in the document. In our case, a query-centric context is a passage of a document from which the most relevant tokens are extracted.

7. Bibliographie

Beltagy I., Peters M. E., Cohan A., “Longformer: The long-document transformer”, *arXiv preprint arXiv:2004.05150*, 2020.

Chang M.-W., Toutanova K., Lee K., Devlin J., “Language model pre-training for hierarchical document representations”, *arXiv preprint arXiv:1901.09128*, 2019.

Devlin J., Chang M.-W., Lee K., Toutanova K., “Bert: Pre-training of deep bidirectional transformers for language understanding”, *arXiv preprint arXiv:1810.04805*, 2018.

Fan Y., Guo J., Lan Y., Xu J., Zhai C., Cheng X., “Modeling Diverse Relevance Patterns in Ad-Hoc Retrieval”, *The 41st International ACM SIGIR Conference on Research Development in Information Retrieval*, SIGIR ’18, Association for Computing Machinery, New York, NY, USA, p. 375–384, 2018.

Guo J., Fan Y., Ai Q., Croft W. B., “A Deep Relevance Matching Model for Ad-Hoc Retrieval”, *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM ’16, Association for Computing Machinery, New York, NY, USA, p. 55–64, 2016.

Hu B., Lu Z., Li H., Chen Q., “Convolutional Neural Network Architectures for Matching Natural Language Sentences”, *in* Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger (eds), *Advances in Neural Information Processing Systems 27*, Curran Associates, Inc., p. 2042-2050, 2014.

Huang P.-S., He X., Gao J., Deng L., Acero A., Heck L., “Learning Deep Structured Semantic Models for Web Search Using Clickthrough Data”, *Proceedings of the 22nd ACM International Conference on Information Knowledge Management*, CIKM ’13, Association for Computing Machinery, New York, NY, USA, p. 2333–2338, 2013.

Li C., Yates A., MacAvaney S., He B., Sun Y., “PARADE: Passage representation aggregation for document reranking”, *arXiv preprint arXiv:2008.09093*, 2020.

Li X., Mao J., Wang C., Liu Y., Zhang M., Ma S., “Teach Machine How to Read: Reading Behavior Inspired Relevance Estimation”, *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR’19, Association for Computing Machinery, New York, NY, USA, p. 795–804, 2019.

Microsoft, *LETOR: Learning to Rank for Information Retrieval*. 2009.

Mitra B., Diaz F., Craswell N., “Learning to Match Using Local and Distributed Representations of Text for Web Search”, *Proceedings of the 26th International Conference on World Wide Web*, WWW ’17, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, p. 1291–1299, 2017.

NIST, *MQ2007 dataset*. 2008.

Pang L., Lan Y., Guo J., Xu J., Xu J., Cheng X., “Deeprank: A new deep architecture for relevance ranking in information retrieval”, *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, p. 257-266, 2017.

Pennington J., Socher R., Manning C. D., “GloVe: Global Vectors for Word Representation”, *Empirical Methods in Natural Language Processing (EMNLP)*, p. 1532-1543, 2014.

Qin T., Liu T.-Y., Xu J., Li H., “LETOR: A Benchmark Collection for Research on Learning to Rank for Information Retrieval”, *Inf. Retr.*, vol. 13, n^{o} 4, p. 346–374, August, 2010.

Severyn A., Moschitti A., “Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks”, *Proceedings of the 38th International ACM SIGIR Conference on Research and*

Development in Information Retrieval, SIGIR '15, Association for Computing Machinery, New York, NY, USA, p. 373–382, 2015.

Shen Y., He X., Gao J., Deng L., Mesnil G., "A Latent Semantic Model with Convolutional-Pooling Structure for Information Retrieval", *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, Association for Computing Machinery, New York, NY, USA, p. 101–110, 2014.

Thonet T., Cinar Y. G., Gaussier E., Li M., Renders J.-M., "SmoothI: Smooth Rank Indicators for Differentiable IR Metrics", *Submitted*, n.d.

Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I., "Attention Is All You Need", *CoRR*, 2017.

Xiong C., Dai Z., Callan J., Liu Z., Power R., "End-to-End Neural Ad-Hoc Ranking with Kernel Pooling", *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, Association for Computing Machinery, New York, NY, USA, p. 55–64, 2017.

Yang Z., Yang D., Dyer C., He X., Smola A., Hovy E., "Hierarchical attention networks for document classification", *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, p. 1480-1489, 2016.

Zhang X., Wei F., Zhou M., "HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization", *arXiv preprint arXiv:1905.06566*, 2019.