Introduction

The models attempt to incorporate position-aware information like proximity and term dependency into document relevance evidence. In SDM and FDM [6], the term proximity of pairs of terms is modeled with Markov random fields which we can break down to three connection types: individual term, ordered and unordered. Inspired by image recognition tasks, MatchPyramid [18] feeds query-document similarity matrices through CNN, pooling and finally two fully connected networks to aggregate the document scores. PACRR [9] feeds query-document similarity matrices through a CNN with different kernel sizes, and then through two max-pooling layers to summarize signals of each query term.Finally the model flattens the n-gram signals and performs LSTM to aggregate the document scores with additional term feature IDF. Building upon KNRM for modeling soft-matching, Conv-KNRM [8] applies CNN on query and document embeddings respectively to extract n-gram level embeddings. The similarities are calculated by cross-matching the n-gram embeddings of query and document, and then are summarized with kernel pooling, and finally aggregated via Learning-to-rank layer.

Position-Aware

Term Dependency [12] , Term Proximity

Both SDM and FDM [6] make use of features functions based on ordered and unordered phrases to model dependency of pairs of terms. The ordered potential function accounts for adjacent sequential matches while the unordered features function captures the signal that the query terms appear within a given window, whether the query terms are in order or not. Though unordered phrases are taken into consideration, unlike FDM, SDM considers dependency only between adjacent terms. For example, a top google query: “youtube to mp3” [20]. In FDM, dependency of “youtube” and “mp3” are also considered which makes it capture documents with desired results like Youtube mp3 converter. In this case, one can assume FDM performs better than SDM. We can see the result (cf. Table 6 in [6]) shows that both SDM and FDM have constant improvement across different data collections on the full independence model.

Since CNN was demonstrated to compose n-gram embeddings efficiently, neural approaches started to take advantage of CNN to deal with multi-term dependency. MatchPyramid [18] takes a kernel size of 1*3 which optimizes the experiment. Convolving with a kernel of 1*3 enables the model to concatenate signals between document terms while not concatenating along the query terms. Thus, the term proximity property is less addressed. The unexpected experiment result might be due to the insufficient data size for training the model to learn the larger number of parameters that bigger kernels introduce.

Conv-KNRM [8] applies 1-D convolution over query and document embeddings respectively. This step allows the model to extract n-gram features into the same dimension and cross-match query-document similarity matrices between n-grams of different lengths. The n-gram feature accounts for adjacent sequential matches. Inheriting from K-NRM, Conv-KNRM weights the different levels of soft-matching features for each term. In K-NRM, IDF weighting is experimented and no positive results are achieved compared to simple aggregation. Thus, no n-gram weighting is conducted in the model.

PACRR [9] extracts n-gram with a CNN over similarity matrix and max-pooling. One may assume that the convolutional kernel is playing the role of window proximity constraints. However, in the following max-pooling layer, only filters accounting for the strongest signal are kept in each kernel size, that is an exact ordered n-grams match. Hence, so far only n-gram adjacent sequential dependency is considered here. The different sizes of convolution kernel are set for n-grams of different lengths. In the last component, LSTM is employed to aggregate the query signals which accounts for the long-term dependencies between query terms. This allows the model to consider ordered non-contiguous proximity in a sophisticated manner so that the model assigns different importance to each term dependency. Compared to SDM or proximity component Tao et al. [13] proposed, the model can handle queries like “KFC fast food franchise” more efficiently, because “KFC” can be detected to be the most important evidence to rank documents. The weighting can also exclude undesired term dependencies. Undesired term dependencies being extracted is an artifact of term proximity, ex: in the query “white house rose garden”, “white house” and “rose garden” are valuable proximity features, yet “white rose” might divert the task to something irrelevant. LSTM component here eliminates the artifact of undesired term dependencies.

Query Coverage

SDM and FDM [6] make use of bag-of-word potential function. In the bag-of-word potential function, the smoothed term frequency is log-scaled before computing the weighted sum. This log-scale mechanism avoids the exaggerated effect of high term frequency and also implicitly makes the model favor more distinct query terms.

MatchPyramid [18] takes pooling size of 3*10 over query-document similarity matrix. The pooling concatenates signals between the nearest query term prior to and followed by the query term. The model does not include salient information of each query term, rather the signal of each term is blurred out along the process.

PACRR [9] aggregates signals of each query term preceded by k-max pooling. The k-max pooling along the query terms allows the model to extract the k-most important signal of each query. Conv-KNRM [8] takes eleven kernels to summarize the signal per query terms and log-sums the extracted soft-tf feature. The kernel pooling allows different levels of soft-tf feature being extracted along the query term. Moreover, Conv-KNRM sets one kernel of $\mu = 1, \sigma = 10^{-3}$ to account for exact matches.

Document Length

Conv-KNRM [8] log-scales soft-tf features of each query term to penalize the longer document and PACRR [9] max-pools the top-two signal along the query terms to avoid favoring long documents. In MatchPyramid [18], pooling size is set to 3*10 while average document length is 477 (cf. Table 1 in [18]). Hence, longer documents are more likely to be summarized out more relevance signals then shorter documents and the model is going to favor longer documents. In SDM [6], the term features is inversely scaled with document length and is parameterized by α_D governing the IDF effect.

Discussion

MatchPyramid uses 2D CNN over similarity matrix which seems to have more power to capture complex term dependency patterns, like unordered proximity. However, the result is worse than PACRR and Conv-KNRM (cf. Table 3 in [8]). As mentioned before, the choice of kernel and pooling size might limit the model potential. Different pooling sizes are explored in the MatchPyramid experiment. It turns out that pooling size 3*10 optimizes the performance and thus is chosen. However, the pooling size might just happen to work better in the specific experiment setting, where the title of the topics of the data are taken as queries, which does not sound general enough for ad-hoc information retrieval tasks. To account for query coverage, a pooling size of like 1*3 might be a more interpretable choice.

Both PACRR and Conv-KNRM extract n-gram with CNN layer. Conv-KNRM applies 1D convolution with filters to extract n-gram features and then cross-match the features to get similarities; PACRR applies 2D convolution on similarity matrix, then max-pool to extract n-gram features. Conv-KNRM summarizes the similarities with kernel-pooling and feeds the query signals to fully connected networks while PACRR summarizes the similarities with max pooling and feeds the query signals to LSTM. In the phrase level embeddings, Conv-KNRM considers unigram while PACRR does not. Alternatively, PACRR adapts LSTM considering ordered non-contiguous proximity which enables the model to include the unigram case. The classroom example “Bayern Munich vs. Liverpool" would be considered in both models for the query “Bayern vs. Liverpool". As discussed above, PACRR weights the n-gram query feature with LSTM incorporating another classic query term feature IDF. Conv-KNRM does not weigh the n-gram embeddings directly but weights the different levels of semantic similarity in learning to rank layers. Thus, PACRR clearly will work better for that query with imbalanced information. However, I would assume Conv-KRNM performs better in the scenario that the user types no redundant word in the query because the extra filters used to extract different levels of soft-matching. Both can be trained end-to-end and this suggests that both models have the potential to adapt pre-trained embeddings to other similar-domain queries. Also, PACRR is more efficient given the number of parameters.

Despite the fact that FDM is such a flexible model for term dependency, especially any arbitrary feature can be incorporated, the dependency relationship is exponential to the query length which limits the application of long queries. The result (cf. Table 6 in [6]) shows that SDM performance is comparable to FDM and even outwin in AP, WSJ. Neural models like PACRR and Conv-KNRM are not robust to rare seen or unseen queries because it is hard to generalize rare terms through embedding relevance signals. SDM on the other hand takes precise retrieval of documents containing the rare terms. Also, SDM can model unordered dependency. However, the result (cf. Table 5 in [6]) shows that combining unordered dependency into the model offers little improvement on the combination of individual term and ordered. Although Donald Metzler and W. Bruce Croft [6] claim that approximations often do not maximize the correct objective function and so they choose to perform parameter sweep to optimize the parameters, this exhausted approach limits the model to capture other important information like query term importance. Despite the fact that the learning to rank layer does not directly maximize mean average precision, it is still a good alternative to consider. With the approximation learning approach, extra weights can be introduced for query terms or proximity distances. To model multi-term dependency, Markov-chain might not be the first option because with Markov-chain longer phrases always have less or equal probability then short phrases. It does not always hold true.

Reference

[6] Donald Metzler and W. Bruce Croft. 2005. A Markov random field model for term dependencies. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '05). [pdf]

[7] Donald Metzler. 2011. A Feature-Centric View of Information Retrieval. [chapter pdfs] (link only works from uni network)

[8] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional Neural Networks for Soft-Matching N-Grams in Ad-hoc Search. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18). [pdf]

[9] Kai Hui, Andrew Yates, Klaus Berberich, Gerard de Melo. 2017. PACRR: A Position-Aware Neural IR Model for Relevance Matching. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP '17). [pdf]

[12] Samuel Huston and W. Bruce Croft. 2014. A Comparison of Retrieval Models using Term Dependencies. In Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM '14). [pdf]

[13] Tao Tao and ChengXiang Zhai. 2007. An exploration of proximity measures in information retrieval. In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07). [pdf]

[14] Chris Olah. 2015. Understanding LSTM Networks. Blog post. [link]

Retrieval. In Proceedings of ECIR 2008. [pdf]

[18] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2016. A Study of MatchPyramid Models on Ad-hoc Retrieval. In the SIGIR 2016 Workshop on Neural Information Retrieval (NeuIR '16). [pdf]

[19] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. 2016. Text Matching as Image Recognition. 2016. [pdf]

[20] Top Google searches (as of May 2019) [link]

#[10] Andrew Yates, Kai Hui. 2017. DE-PACRR: Exploring Layers Inside the PACRR Model. In the SIGIR 2017 Workshop on Neural Information Retrieval (NeuIR '17). [pdf]

#[11] Kai Hui, Andrew Yates, Klaus Berberich, Gerard de Melo. 2018. Co-PACRR: A Context-Aware Neural IR Model for Ad-hoc Retrieval. In Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18). [pdf]

#[15] Yixing Fan, Jiafeng Guo, Yanyan Lan, Jun Xu, Chengxiang Zhai, and Xueqi Cheng. 2018. Modeling Diverse Relevance Patterns in Ad-hoc Retrieval. In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18). [pdf]

#[16] Zhiwen Tang and Grace Hui Yang. 2019. DeepTileBars: Visualizing Term Distribution for Neural Information Retrieval. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19). [pdf]

#[17] Michael Bendersky and Oren Kurland. 2008. Utilizing Passage-Based Language Models for Document