Consistency and Variation in Kernel Neural Ranking Model

Mary Arpita Pyreddy*
Carnegie Mellon University
mpyreddy@cs.cmu.edu

Varshini Ramaseshan*
Carnegie Mellon University
vramases@cs.cmu.edu

Narendra Nath Joshi*
Carnegie Mellon University
nnj@cs.cmu.edu

Zhuyun Dai
Carnegie Mellon University
zhuyund@cs.cmu.edu

Chenyan Xiong
Carnegie Mellon University
cx@cs.cmu.edu

Jamie Callan
Carnegie Mellon University
callan@cs.cmu.edu

Zhiyuan Liu
Tsinghua University
liuzy@tsinghua.edu.cn

ABSTRACT

This paper studies the consistency of the kernel-based neural ranking model (K-NRM), a recent state-of-the-art neural IR model, which is important for reproducible research and deployment in the industry. We find that K-NRM has low variance on relevance-based metrics across experimental trials. In spite of this low variance in overall performance, different trials produce different document rankings for individual queries. The main source of variance in our experiments was found to be different latent matching patterns captured by K-NRM. In the IR-customized word embeddings learned by K-NRM, the query-document word pairs follow two different matching patterns that are equally effective, but align word pairs differently in the embedding space. The different latent matching patterns enable a simple yet effective approach to construct ensemble rankers, which improve K-NRM's effectiveness and generalization abilities.

KEYWORDS

Neural-IR, Retrieval Model Stability, Ensemble-Rankers

ACM Reference Format:
Mary Arpita Pyreddy, Varshini Ramaseshan[1], Narendra Nath Joshi[1], Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Consistency and Variation in Kernel Neural Ranking Model. In *SIGIR '18: 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, July 8-12, 2018, Ann Arbor, MI, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3209978.3210107

1 INTRODUCTION

Neural IR models have received much attention due to their continuous text representations, soft-matching of terms, and sophisticated non-linear models. However, the non-convexity and stochastic training of neural IR models raises questions about their consistency compared to heuristic and learning-to-rank models that use discrete representations and simpler methods of combining evidence. Consistent behavior under slightly different conditions is essential to reproducible research and deployment in industry.

This paper studies the stability of K-NRM, a recent state-of-the-art neural ranking model [10]. K-NRM learns the word embeddings and ranking model from relevance signals. Its effectiveness is due to word embeddings tailored for search tasks and kernels that group matches into bins of different quality.

To better understand its stability, we compare the behavior of multiple trained models under similar conditions. We find that although K-NRM produces similar accuracy across different trials, it also produces rather different document rankings for individual queries.

Analysis of weights learned for K-NRM kernel scores (soft-match features) revealed that weights from different trials match one of two patterns. The word embeddings reflect these patterns. Trials whose kernel weights have the same pattern have similar word embeddings. Interestingly, the two patterns are equally effective.

The difference in the ranking patterns from different K-NRM trials makes them a good fit for ensembles. Aggregating scores from different trials enables an ensemble to promote documents that multiple trials agree are most likely to be relevant. Experimental results show that simple K-NRM ensembles significantly boost its ranking accuracy and improve its generalization ability.

2 RELATED WORK

Recent neural IR methods can be categorized as *representation-based* and *interaction-based* [2]. *Representation-based* models use distributed representations of the query and document that are matched in the representation space [4, 8]. *Interaction-based models* use local interactions between the query and document words and neural networks that learn matching patterns [2, 10].

K-NRM [10] is an interaction-based model that uses kernel pooling to summarize word-word interactions. It builds a word-word similarity matrix from word embeddings, and uses kernel pooling to 'count' the soft matches at multiple similarity levels using Gaussian kernels. A linear learning-to-rank layer combines the kernel features. The whole model is end-to-end trainable. When trained from a search log, K-NRM outperforms neural IR methods and feature-based learning-to-rank methods.

*The first three authors contributed equally

SIGIR '18, July 8–12, 2018, Ann Arbor, MI, USA

ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210107

Most neural IR research focuses on ranking accuracy. However, the high variance of deep learning models causes concern about their consistency. Haber et al. [3] identify the causes for lack of stability and high variance as the dimensionality and non-convexity of the optimization problem. A common method to reduce variance and improve generalization is to create an ensemble of models [6]. Krogh and Vedelsby [6] argue that a good ensemble is one where the components are all accurate but disagree on individual examples. Ensembles of neural network have been applied successfully to tasks such as image classification [5] and machine translation [9].

3 EXPERIMENTAL SETUP

Our experiments followed the original K-NRM work [10] and used its open-source implementation[1]. We used the same click log data from Sogou.com, a Chinese web search engine. The training set contained 95M queries, each with 12 candidate documents on average. The testing set contained 1,000 queries, each with 30 candidate documents on average. Documents were represented by titles. Xiong, et al. [10] built the vocabulary from queries and titles, but we built it from the queries, titles and URLs for better term coverage.

Training Labels: The relevance labels for training were generated by the DCTR [1] click model from user clicks in the training sessions. DCTR uses the clickthrough rate for each query-document pair as the relevance score.

Testing Labels: Following Xiong et al. [10], three testing conditions were used. **Testing-SAME** used DCTR to generate the testing labels while **Testing-DIFF** employed a more sophisticated model, TACM [7]. TACM takes into account both clicks and dwell times to generate testing labels. **Testing-RAW** treated the only clicked document in each single-clicked session as a relevant document, and used MRR (Mean Reciprocal Rank) as the metric. Testing-DIFF and Testing-RAW were considered more reliable than Testing-SAME because they are less subject to over-fitting.

Model Configuration: We adopted the same default hyper-parameter configuration and 11 Gaussian kernels as in prior work [10]. The first kernel had $\mu = 1, \sigma = 10^{-3}$ to cover exact matches. The other 10 kernels were equally split in the cosine value range $[-1, 1]$: $\mu_1 = 0.9, \mu_2 = 0.7, ..., \mu_{10} = -0.9$; σ was set to 0.1. Word embeddings were initialized with a pretrained word2vec. The model used the Adam optimizer and was trained with batch size 16, learning rate = 0.001, and $\epsilon = 1e - 5$. The order of training data batches was fixed. An early stopping condition of 2 epochs was used in all experiments.

The model was implemented using TensorFlow. All experiments were executed on a p2.xlarge AWS instance with 4 virtual CPUs and 1 NVIDIAGK210 GPU.

4 VARIANCE

The first experiment studied the consistency of K-NRM by running 50 stochastically trained models with random initialization. The consistency among the 50 trials is examined at the query-set level and the individual query level.

The performance of 50 models on three metrics is summarized in Table 1. The min/max differences are large, especially for NDCG@1. However the standard deviations are small, ranging from 0.5-1.3% absolute, and 1-4% relative to mean values. The min/max differences are due to a small number of outliers. Performance is stable across most trials. Table 1 also shows results reported by Xiong et al [10]. Their model performance falls in the lower end of our trials, probably due to different vocabularies and stopping conditions.

The next analysis studied the consistency at the individual query level by examining document rankings generated by different trials. For each query we examined the top k ranked document from 10 different trials. The total number of distinct documents indicates how well the models agree about which documents to place at the top of the ranking. A histogram (Figure 1) shows the number of queries at each agreement level for top 1, 3, and 10 ranked documents.

Different trials rank different documents at the top to some extent. For about 50 of the 1000 queries, all 10 trials select the same document at rank 1 (Figure 1a); for 35% of the queries, the trials select 2-3 different documents. Moderate consistency is observed across the trials. Only 15% of the queries get more than 5 different documents from the 10 trials. None of the queries get 10 completely different documents at the top 1.

Figure 1b shows a similar trend. For 66% of the queries, the 10 trials collectively select 3-9 different documents for the top 3 slots. The document sets from the 10 trials converge deeper in the rankings. In the top 10 slots (Figure 1c), the histogram shifts left, indicating that the 10 trials have higher agreement. This is expected because K-NRM re-ranks the top 30 documents. The disagreement in the top 1 and 3 ranks indicates that although different trials have similar sets of documents, their rankings are slightly different.

5 LATENT MATCHING PATTERNS

To better understand the model differences, we investigated the model parameters through multiple K-NRM trials. K-NRM has two trainable components: the word embedding layer and the learning-to-rank layer. The word embedding layer aligns query-document word pairs and assigns them to the closest kernels by their cosine similarity. The learning-to-rank layer learns the importance of word pairs around each kernel. This analysis studied both parameters.

Figure 2 plots the learning-to-rank weights from 10 random trials. The trials fall into two main patterns. One pattern starts with a downward slope and then moves upward while the second pattern goes the other way. K-NRM allocates word pairs into kernels based on their contribution to relevance. Different learning-to-rank weights indicate different ways of allocating word pairs to kernels.

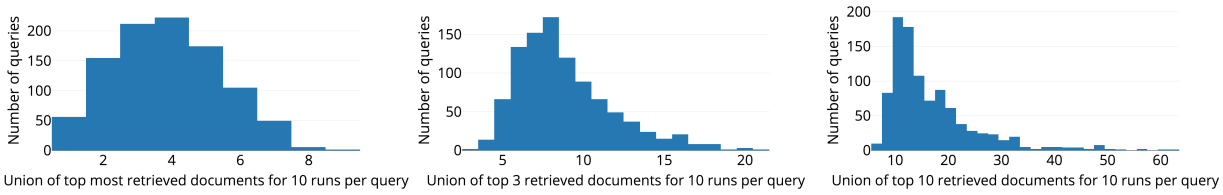
We further studied the two patterns with word embeddings from multiple trials. We randomly picked 5 runs. Runs A1–A3 belonged to one learning-to-rank weight pattern (Pattern A); runs B1–B2 belonged to the other pattern (Pattern B). We compared the word pair distribution between pairs of runs through a heat map with each cell (μ_x, μ_y) indicating the fraction of word pairs that fall into kernel μ_x in one run and kernel μ_y in the other run. Figure 3 shows the heat maps between Run A1 and the rest of runs.

Runs from the same pattern have similar heat maps. As shown in Figure 3a and 3b, Runs A2 and A3 show a strong diagonal pattern, indicating that most of the word pairs are in the same kernel as in run A1. Runs from pattern B share another word pair distribution. As can be seen from Figure 3c and 3d, a lot of word pairs are assigned to a different kernel by runs B1/B2 as compared to the kernel

[1] https://github.com/AdeDZY/K-NRM

Table 1: Statistics from 50 K-NRM trials trained with random parameter initialization. Minimum, Mean, and Maximum are the worst, average, and best performances. Standard Deviation is calculated on the corresponding 50 evaluation scores.

	Testing-SAME			Testing-DIFF			Testing-RAW
Statistic	**NDCG@1**	**NDCG@3**	**NDCG@10**	**NDCG@1**	**NDCG@3**	**NDCG@10**	**MRR**
Minimum	0.2531	0.3352	0.4221	0.2983	0.3234	0.4257	0.3430
Mean	0.2859	0.3495	0.4396	0.3242	0.3365	0.4378	0.3547
Maximum	0.3166	0.3744	0.4577	0.3484	0.3532	0.4496	0.3702
Standard Deviation	0.0130	0.0096	0.0104	0.0108	0.0076	0.0052	0.0067
Reported in Xiong, et al.[10]	0.2642	0.3210	0.4277	0.2984	0.3092	0.4201	0.3379



(a) Agreement on Top 1 **(b) Agreement on Top 3** **(c) Agreement on Top 10**

Figure 1: Query level ranking agreement. The X-axes are the number of distinct documents that appeared in the top K ranking results of 10 K-NRM trials. Larger X values indicate lower agreement among trials.

Table 2: The performance of ensemble models using different methods for selecting base models. K-NRM Mean is the average performance of 50 base models. $*, \dagger, \S$ indicate statistically significant improvements ($p < 0.05$) over K-NRM Mean, Ensemble-A and Ensemble-B respectively.

	Testing-SAME			Testing-DIFF			Testing-RAW
Model	**NDCG@1**	**NDCG@3**	**NDCG@10**	**NDCG@1**	**NDCG@3**	**NDCG@10**	**MRR**
K-NRM Mean	0.2859	0.3495	0.4396	0.3242	0.3365	0.4378	0.3547
Ensemble-A	0.3270 (14%)$*$	0.3824 (9%)$*$	0.4691 (7%)$*$	0.3702 (14%)$*$	0.3694 (10%)$*$	0.4573 (4%)$*$	0.3908 (10%)$*$
Ensemble-B	0.3215 (12%)$*$	0.3723 (7%)$*$	0.4570 (4%)$*$	0.3831 (18%)$*$	0.3749 (11%)$*$	0.4629 (6%)$*$	0.3930 (11%)$*$
Ensemble-A&B	0.3359 (17%)$*\dagger\S$	0.3811 (9%)$*\dagger\S$	0.4689 (7%)$*\dagger\S$	0.3931 (21%)$*\dagger\S$	0.3841 (14%)$*\dagger\S$	0.4684 (7%)$*\dagger\S$	0.4035 (14%)$*\dagger\S$

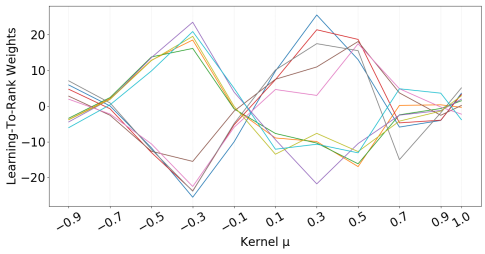


Figure 2: Learning to rank weights from 10 K-NRM trials. The X axis is the μ of a kernel. The Y axis is its ranking weight.

assigned by run A1. The results reveal two distinct latent matching patterns. Trials from the same pattern have similar learning-to-rank weights and word embeddings. The two patterns differ largely in their word pair alignment.

Although the two patterns align word embeddings differently, both are equally effective and produce similar accuracy (Table 1).

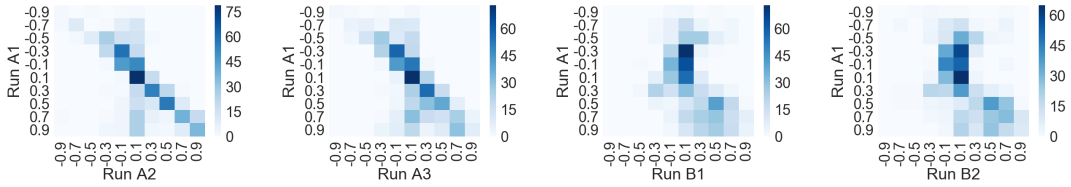
6 ENSEMBLE MODEL

The different rankings and distinct patterns in multiple K-NRM trials provided possibilities to reduce risk and improve the model's generalization ability using ensemble models [6]. The following experiments studied the effectiveness of ensemble models.

We used an unweighted-average ensemble model [5] that averages the scores from multiple trials. To investigate the effects of latent matching patterns, we tested different types of ensemble models: Ensemble-A used 10 base models randomly selected from Pattern A; Ensemble-B used 10 base models from Pattern B; Ensemble-A&B used base models from both patterns, 5 from each[2]. To make evaluation reliable, 10 ensemble rankers were generated for each method with different base models randomly chosen from a pool of 50 K-NRM trials.

All ensemble methods significantly outperformed individual models (Table 2). The differences in document rankings allowed multiple trials to 'vote' in the ensemble model. Documents favored by the majority of trials are voted up, whereas documents that are

[2]We found that performance saturates with more than 10 base models.



(a) Run A1 vs. Run A2 (b) Run A1 vs. Run A3 (c) Run A1 vs. Run B1 (d) Run A1 vs. Run B2

Figure 3: Word pair movements between runs from two patterns, A and B. Each cell (μ_x, μ_y) in the heat map indicates the number of word pairs whose cosine similarities fall into Kernel μ_y in Run A1 (Y-axis) and kernel μ_x in the other run (X-axis).

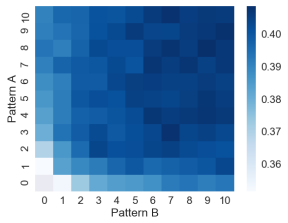


Figure 4: The accuracy of ensemble models that combine different numbers of base models from Patterns A and B. Each cell is the MRR (Testing-RAW) of an ensemble model built with m Pattern A models and n Pattern B models.

wrongly ranked high in a poor trial are voted down. Comparing NDCG scores at different depths, we see that ensembles are most effective at the top of the ranking. This is because the dataset mostly contains 20-30 documents per query. There is more opportunity for disagreement at the top, which gives more scope for improvement. The ensemble generalization ability is reflected by the improved performance on Testing-DIFF and Testing-RAW as compared to Testing-SAME (Table 2).

`Ensemble-A&B` outperformed `Ensemble-A` and `Ensemble-B` (Table 2), which indicates that having and recognizing two distinctive matching patterns is beneficial to ensemble models.

To further understand the effects of the two patterns, we tested ensemble models with `m` Pattern A models and `n` Pattern B models. Figure 4 shows MRR on Testing-RAW as a heatmap. It confirms that having two variations enables better ensembles; ensemble models that only used one pattern have the lowest accuracy. Compared to single pattern ensembles, mixed ensembles can achieve the same accuracy using a smaller ensemble model with fewer base models. For example, cell $(3, 3)$ has higher accuracy than cell $(10, 0)$. Besides, ensemble models benefit from a balanced mix of the two patterns, as seen from the darker cells around the diagonal which have similar number of base trials from each pattern.

Prior research did not recognize that `K-NRM` consistently converges to a small number of distinct, equally-good local optima. Recognizing this helps in constructing high-quality ensembles.

7 CONCLUSION

This paper studies the consistency and variation of `K-NRM`, a recent state-of-the-art neural ranking model. Unlike feature-based methods where features are stable and ranking models are often convex, neural networks are non-convex and employ stochastic training, making it important to consider the ranking stability in neural IR. By investigating multiple trials of `K-NRM`, we find that its accuracy is quite stable (has low standard deviation) in spite of its random components. However, stable NDCG does not imply identical rankings at the individual query level. Different trials have moderate agreement about which document to rank first. Ten trials collectively select 1-3 documents to rank first for 40% of our queries.

Our analyses further demonstrate that multiple trials of `K-NRM` converge to two latent patterns that are about equally effective. Runs within the same pattern converge to similar ranking weights and word embeddings. This behavior was not recognized by prior work, and is worth additional study.

The distinct but equally effective matching patterns makes `K-NRM` a good fit for ensemble models. Recognizing different convergence patterns and selecting ensemble components equally from each pattern further improves `K-NRM`'s accuracy and ability to generalize.

8 ACKNOWLEDGMENTS

This research was supported by National Science Foundation (NSF) grant IIS-1422676. Any opinions, findings, and conclusions are the authors' and do not necessarily reflect those of the sponsor.

REFERENCES

[1] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. Click Models for Web Search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* (2015), 1–115.
[2] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM CIKM*. ACM, 55–64.
[3] Eldad Haber and Lars Ruthotto. 2017. Stable architectures for deep neural networks. *Inverse Problems* 34, 1 (2017), 014004.
[4] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM CIKM*. ACM, 2333–2338.
[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in NIPS*.
[6] Anders Krogh and Jesper Vedelsby. 1995. Neural network ensembles, cross validation, and active learning. In *Advances in NIPS*. 231–238.
[7] Yiqun Liu, Xiaohui Xie, Chao Wang, Jian-Yun Nie, Min Zhang, and Shaoping Ma. 2016. Time-Aware Click Model. *ACM TOIS* (2016), 16.
[8] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM CIKM*. ACM, 101–110.
[9] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in NIPS*. 3104–3112.
[10] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of ACM SIGIR 2017*. ACM, 55–64.