

Explicit and Implicit Semantic Ranking Framework

Xiaofeng Zhu
xiaofzhu@microsoft.com
Microsoft
Redmond, WA, USA

Thomas Lin
tlin@microsoft.com
Microsoft
Redmond, WA, USA

Vishal Anand
visanand@microsoft.com
Microsoft
Redmond, WA, USA

Matthew Calderwood
matthew.calderwood@nuance.com
Nuance Communications
Burlington, MA, USA

Eric Clausen-Brown
erclaus@microsoft.com
Microsoft
Redmond, WA, USA

Gord Lueck
gordonl@microsoft.com
Microsoft
Redmond, WA, USA

Wen-wai Yim
yimwenwai@microsoft.com
Microsoft
Redmond, WA, USA

Cheng Wu
wucheng@microsoft.com
Microsoft
Redmond, WA, USA

ABSTRACT

The core challenge in numerous real-world applications is to match an inquiry to the best document from a mutable and finite set of candidates. Existing industry solutions, especially latency-constrained services, often rely on similarity algorithms that sacrifice quality for speed. In this paper we introduce a generic semantic learning-to-rank framework, Self-training Semantic Cross-attention Ranking (*sRank*). This transformer-based framework uses linear pairwise loss with mutable training batch sizes and achieves quality gains and high efficiency, and has been applied effectively to show gains on two industry tasks at Microsoft over real-world large-scale data sets: Smart Reply (SR) and Ambient Clinical Intelligence (ACI). In Smart Reply, *sRank* assists live customers with technical support by selecting the best reply from predefined solutions based on consumer and support agent messages. It achieves 11.7% gain in offline top-one accuracy on the SR task over the previous system, and has enabled 38.7% time reduction in composing messages in telemetry recorded since its general release in January 2021. In the ACI task, *sRank* selects relevant historical physician templates that serve as guidance for a text summarization model to generate higher quality medical notes. It achieves 35.5% top-one accuracy gain, along with 46% relative ROUGE-L gain in generated medical notes.

CCS CONCEPTS

• Information systems → Learning to rank; • Computing methodologies → Learning to rank.

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. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

WWW '23 Companion, April 30-May 4, 2023, Austin, TX, USA

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9419-2/23/04...\$15.00
<https://doi.org/10.1145/3543873.3584621>

KEYWORDS

semantic search, pairwise learning to rank, smart reply, text summarization, transformer, dual encoder

ACM Reference Format:

Xiaofeng Zhu, Thomas Lin, Vishal Anand, Matthew Calderwood, Eric Clausen-Brown, Gord Lueck, Wen-wai Yim, and Cheng Wu. 2023. Explicit and Implicit Semantic Ranking Framework. In *Companion Proceedings of the ACM Web Conference 2023 (WWW '23 Companion)*, April 30-May 4, 2023, Austin, TX, USA. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3543873.3584621>

1 INTRODUCTION

Learning-to-rank frameworks are versatile and extensible, especially in production environments where classification fails to scale or reaches performance limitations. In explicit information retrieval systems such as search engines, the design specifics of these frameworks are important - e.g., training and inference setup, data, and learning-to-rank model loss functions. In this paper we present a learning-to-rank framework *sRank* developed for and tested on two industry tasks which both contain retrieval components with one binary relevance: Smart Reply (SR) and Ambient Clinical Intelligence (ACI). *sRank* is tailored toward optimization and generalization to meet production requirements for these tasks.

With the popularity of deep neural ranking models in learning-to-rank [10, 22], contextual featurization using Transformer models such as BERT [5] and Big Bird [28] largely eliminate human efforts and achieve high performance in ColBERT [13], PARADE [15], MVA [29], and more [8, 12, 17]. To reduce inference cost for ranking, multi-stage retrieval systems can be set up where the top k documents are first identified using ranking functions like BM25 [19], and then neural models re-rank those k documents. It is also common for production retrieval systems to use CPU for inference with relatively lightweight and less effective models, such as similarity of embedding spaces [12]. However, these studies cannot fully address retrieval challenges for our SR and ACI tasks.

Smart Reply is a system for Microsoft's customer technical support chat that efficiently suggests reply messages for support agents serving multiple products. Its goal is to improve agent productivity,

improve customer satisfaction, and reduce operation costs. Prefabricated/canned reply messages are created and reviewed by agent support specialists. Smart Reply then monitors conversations between customers and support agents, and suggests the top one canned reply message that agents can quickly use when the conversation context relates to a prepared reply. Product requirements for SR include that it must present suggestions faster than the agent’s normal response and search time, it must suggest replies with low error tolerance, and it must provide smart replies only when needed to avoid overwhelming agents.

In the Ambient Clinical Intelligence task, visits between patients and physicians are recorded, transcribed by ASR systems, and then text generation models use this to automatically generate the needed medical note documentation for the encounter [6, 14, 18]. Generating medical documentation for physicians allows physicians to provide more attentive care to their patients, instead of being distracted by note taking and documentation during the encounter. It reduces physician burnout by saving physicians from many hours of documentation work. For medical documentation, physicians often re-use *templates* that they have prepared beforehand for various encounter types. If a ranking model can select which one of the physician’s templates is most appropriate for each new encounter transcript, then this information can be used to guide more accurate medical note generation. Product requirements for ranking in ACI include that it must select the correct template from a set of existing templates, and it must be computationally efficient both for training and inference.

Our *sRank* framework addresses the challenges for SR and ACI. It effectively and efficiently applies a dual-encoder-style cross-attention architecture to learning-to-rank in both training and real-time prediction utilizing document embedding caches and self-training. We show a training technique that enables training over candidate sets of various sizes, and present an efficient method for making pairwise cross entropy linear for our applications. We also explain how *sRank* is able to return no result when there are no correct matches. The primary contributions of this paper are as follows:

- We present *sRank*, an efficient self-training cross-attention learning-to-rank model that can be used for real-time applications, various loss functions, and is scalable to mutable batch sizes. Although we focus on binary relevance applications in this paper, it can also be applied to multi-level ranking or generic contrastive learning, such as Siamese networks [20, 24] or triplet loss [9].
- We demonstrate that pairwise cross-entropy for one binary relevance is $O(n)$ time complexity using tensor calculation, while general RankNet [1] loss is $O(n^2)$ with a reduction to batch-level $O(n^2)$ in an open source learning-to-rank framework TFR-BERT [8]. We reduce inference complexity by caching document embeddings thus self-train and update the embeddings during training.
- We present our ranking components optimized for the real world SR and ACI industry applications. We show 11.7% to 35.5% gain in top-one accuracy, as well as corresponding downstream application gains. These components can also be easily extended to additional industry applications.

We launched an English release of Smart Reply to global regions in January 2021. We report relative improvement metrics over the previous system from 4-months of A/B testing conducted in 2020, and report online metrics from the recent 3-month period in 2022. For ACI, we began ranking research in July 2021 and the work is ongoing. For reasons including to protect customer privacy, this paper reports on relative metric improvements for both applications.

2 BACKGROUND AND RELATED WORK

2.1 Classical Learning-to-rank

Learning-to-rank over classical and general retrieval systems with multi-level relevance (e.g., 0-5 with 0 being irrelevant and 5 being most relevant) often favors listwise loss functions over pairwise loss functions [3, 26, 27]. Listwise loss functions are also chosen over pairwise loss functions for efficiency reasons, especially in neural networks, due to their $O(n)$ calculation instead of $O(n^2)$. However, for the retrieval components of our applications there is only one correct document per set of candidate documents. In Section 3.2 we show how tensor-based pairwise loss calculation can be optimized to $O(n)$ for our use cases.

Let $f(q, D^q)$ be a ranking function for ranking query q and its associated candidate document set D^q (we omit q for simplicity in later sections). As there is only one correct answer for each set of candidate documents, $D^q = d^+UD^-$, where d^+ is the document with label relevance of 1, and D^- indicates the rest of the candidate document with label relevance of 0. NDCG differences in the gradients of LambdaRank and LambdaMart [2] or the loss functions of related work such as [23, 25, 26, 30] become equivalent. We showcase representative RankNet and MLE-based loss functions that maximize the log likelihood of $P(d^+)$ in Equations 1 and 2 respectively.

$$P_t(d^+) = \frac{1}{|\{D^-\}|} \sum_{d^- \in \{D^-\}} \frac{1}{1 + e^{-(f(q,d^+) - f(q,d^-))}}. \quad (1)$$

$$P_t(d^+) = \frac{1}{1 + \sum_{d^- \in \{D^-\}} e^{-(f(q,d^+) - f(q,d^-))}}. \quad (2)$$

The RankNet loss in Equation 1 is less likely to suffer gradient vanishing, and it can be implemented in $O(n)$ instead of batch-level $O(n^2)$ [8] or strict $O(n^2)$ [12]. In addition, current re-ranking models generally require truncating or padding the candidate set size to obtain a universal batch size for training. Section 3.2 shows how we can train and calculate loss effectively for complete sets of documents with varying sizes using mutable batch sizes.

2.2 Transformer-based Re-ranking

Prior studies such as DPR [12] and PreTT [17] define in-batch positive examples explicitly, while they draw in-batch negative examples from a larger pool in the training set, i.e., same negative examples are duplicated in different batches. In addition, training batches all have the same size. Unlike those studies, *sRank* has both an explicit positive example and explicit negative examples in a given batch, resulting in variably sized batches. *sRank* thus requires fewer training resources while obtaining a better pairwise training objective. We propose dual-encoder fashion cross-attention *sRank* that can be executed efficiently in real-time in Section 3.1.

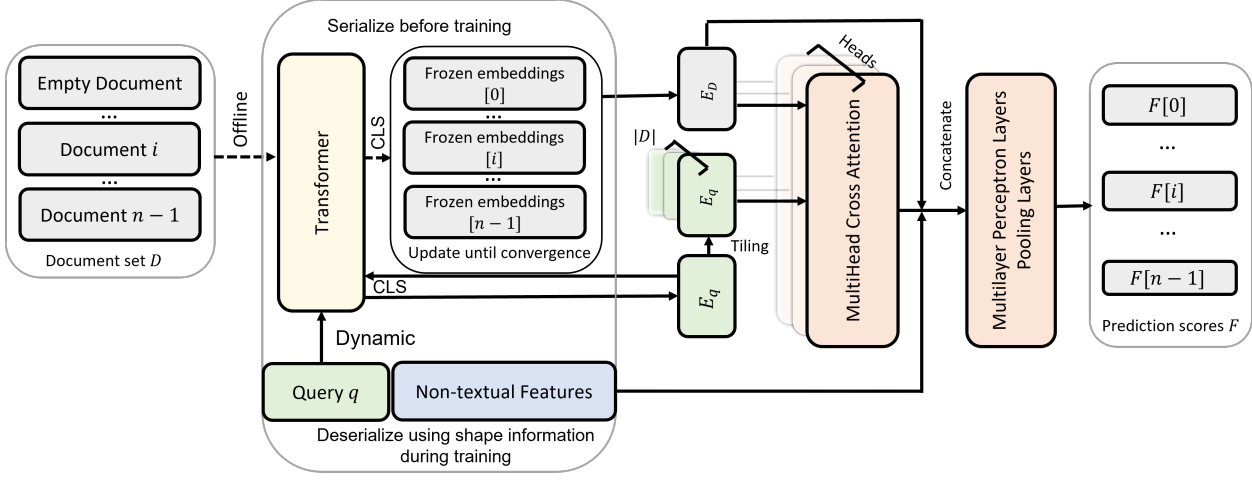


Figure 1: The sRank model takes the serialized record of a query and the frozen embeddings of all its candidate documents before training, deserializes for Multi-Head Cross Attention, and generates the prediction scores F for the candidate documents.

2.3 Application-Specific Requirements

Support agent replies in Smart Reply are either generic and relevant for all products, or apply only for specific products. Clustering-based [11] and classification-based solutions do not satisfy product quality and scalability requirements for this scenario where answer sets may change or grow in this manner.

For the ACI template ranking task, physicians each have their own sets of historical templates, and do not choose templates from other physicians. Physicians can also have different numbers of templates, and the templates can be of varying sizes. Traditional re-ranking approaches are not ideal for this scenario because training batch size is fixed in neural models. A common solution is to truncate or pad candidate documents to a universal batch size in training. We show a generic way of training neural systems on batches of specific sets of candidates of mutable sizes.

Lastly, relevance labels are binary for both SR and ACI, meaning that at most one reply or template can be presented. In this setting, pairwise sigmoid cross entropy is more suitable than listwise loss functions.

3 METHODOLOGY

3.1 Semantic Cross Attention Ranking

The *query-key-value* architecture of Multi-Head Cross Attention represented as (E_q, E_D, E_D) in Figure 1 matches neatly with the expectations of text-based learning-to-rank studies by taking query embeddings/features E_q as *query* and document embeddings E_D as *key* and *value*. We cache the embeddings of candidate documents [7] in the online system to reduce the inference latency from dual encoder with cross-attention. To ensure the inference approach works smoothly, we freeze the embeddings during training.

Instead of costlier methods such as training the embeddings on both queries and candidate documents or pre-training transformers using customized data, we utilize the learning-to-rank training process to update the weights in the transformer to generate more

informative embeddings for the candidate documents. The embeddings of candidate documents are updated after several training epochs (e.g., 10) until the model converges. We apply multi-head cross attention to the frozen embeddings of candidate documents E_D and the dynamic embeddings of each query E_q , tiled to $|D|$.

The batch in sRank contains a serialized record of one question, and embeddings of its candidate documents and feature shapes using Parquet, enabling processing and training of candidate sets with mutable sizes. The negative examples come only from each Microsoft product’s reply set in the SR task and each individual physician’s template set in the ACI task. Negative examples are not duplicated, in contrast to DPR [12]. Questions are not duplicated, in contrast to TFR-BERT [8]. Therefore, our model saves training data and eliminates unnecessary computation of passing data to Transformer. We apply ONNX [4] quantization to inference.

Algorithm 1 Linear pairwise loss for one correct document

- 1: **Input:** labels $Y = (Y_i)_{i=1}^n$
 - 2: **Input:** prediction scores $F = f(q, (d_i)_{i=1}^n)$
 - 3: $P_DIFF \leftarrow F - F^T$
 - 4: $L_DIFF \leftarrow P_DIFF \cdot Y$
 - 5: $S \leftarrow \exp(L_DIFF)$
 - 6: $loss \leftarrow -\frac{1}{n-1} \sum ((1-Y) \odot \ln \frac{1}{1+S})$
 - 7: $loss \leftarrow -\frac{\ln 2 + \sum \ln(1+S)}{n-1}$ \triangleright only one correct document in Y
 - 8: **Return** $loss$
-

3.2 Optimized Loss for Binary Relevance

Algorithm 1 shows the $O(n)$ tensor-based pairwise loss calculation optimized for our use cases. Matrix P_DIFF of size $n \times n$ contains the pairwise prediction score differences and vector L_DIFF of size $n \times 1$ contains the linear score differences between all of the candidate documents and the correct document. We can split candidate documents into multiple batches with the correct document in each batch when n is too large for GPU memory.

4 USE CASES AND EXPERIMENTS

This section describes our experiments and measurements for ranking in SR and ACI. Our primary metric in offline evaluation is top-one accuracy because we suggest at most one reply message for SR and at most one physician template for ACI. The proposed loss is 2-7% better than the MLE loss from [30] on the two tasks.

4.1 Smart Reply for Customer Support

The Smart Reply task is to take the most recent support agent message and the most recent customer message in a customer support conversation, and choose the best reply from a set of canned reply templates. An example Smart Reply could be "This [link](#) has step-by-step instructions for how to activate Microsoft 365." An efficient CPU-based classifier is applied at runtime to classify which specific product an incoming support message interaction is for, and then our learning-to-rank model is used to select the best reply from the canned replies for that product. Smart Reply is able to support customer support interactions across 22 Microsoft products such as Office 365, Teams, Surface, and Remote Assistance.

Table 1 describes statistics for the Smart Reply task. Training and test data were based on an 80%:20% split ratio. Customized tokenization was first applied to message pairs and canned replies, and then DistilBERT [21] was used to vectorize the queries and documents for ranking. One difference from traditional retrieval systems which always retrieve the top-k documents is that we do not want to overwhelm support agents with replies when there are no good canned replies for a customer message. To achieve this, we added a "Silent" class in the product classifier and an "Empty" canned reply in each candidate reply set. We used data augmentation to generate synthetic conversations. For instance, appending a message pair that returns the "Silent" class or "Empty" reply to non-empty questions enriched non-empty triplets, and only using agent or customer messages further enlarged the data size.

Cleaned customer-agent message pairs	1.3 Million
Maximum input tokens	512
Canned reply templates	200
Supported Microsoft products	22
Canned reply templates per product	3-26
Data set size with augmentation	10 Million

Table 1: SR data statistics

Top-one accuracy gains	11.7
Click-through rate (CTR) uplift	42.5
Agent satisfaction improvement	13.4
Time reduction for composing agent messages	38.7

Table 2: SR offline and online metric gains (%)

Table 2 shows the 11.7% offline top-one accuracy gain of sRank compared to our previous DSSM-based [10] system that took transformer embeddings as inputs. We exposed Smart Reply to insider agents for initial feedback then to 50% global agents during A/B testing. sRank also increased CTR on Smart Replies by an absolute 42.5%. During A/B testing, Smart Reply with sRank led to 13.4% increase in agent satisfaction compared to the group not using Smart Reply, and agents composed replies 38.7% faster with Smart Reply.

4.2 Template Ranking in ACI

The second industry task we tackle is Template Ranking in Ambient Clinical Intelligence. Physicians can have sets of templates that they

start from when composing medical documentation. An example of a template could be: "General Appearance: Height __ inches. Weight __ pounds. The patient is alert and oriented and in no distress." If a template ranking system can automatically select the best template to use for an encounter, then this can be used to guide the automatic generation of a more accurate AI medical note.

Table 3 describes statistics for the ACI task. For this task the query is the medical encounter transcript and the candidate documents are physician's templates plus an "Empty" template to represent if no template should be used. We utilized Big Bird RoBERTa to generate conversation and template embeddings. Once our ranking model selected a template for an encounter, the template and the encounter transcript were concatenated and passed to the note generation model to generate the medical note. We focused on guiding generation of the Physical Exam section of Orthopedics clinical notes, because this section often employs templates. The baseline system is a DPR ranking model which meets the inference time requirements for this task.

Medical encounters for ranker training	1 Million+
Maximum input tokens	4096
Total number of medical templates	6118
Medical templates per physician	8-39

Table 3: ACI template modeling configuration

ROUGE-L of baseline relative to ROUGE-L with oracle templates	46.0
ROUGE-L of sRank relative to ROUGE-L with oracle templates	92.0
Top-one accuracy gain over DPR	35.5
Top-one accuracy gain (<25% new templates)	41.5
Top-one accuracy gain (25-75% new templates)	40.6
Top-one accuracy gain (>75% new templates)	20.7

Table 4: ACI sRank metric gains (%)

When medical note generation is guided by the correct template that the physician would use (the "oracle" template), this significantly increases the quality of generated medical notes. The challenge is then how to predict the correct template at run time. Note generation with no template guidance achieved only 46% of the ROUGE-L [16] of using oracle guidance. The DPR model was unable to predict templates at sufficient accuracy, and using its templates led to 2% ROUGE-L drop compared to using no templates. sRank predicted templates more accurately, and end-to-end ROUGE-L with sRank was 92% of oracle ROUGE-L. This highlighted the ability of sRank to effectively guide the generation of higher quality medical notes. Table 4 shows sRank metrics for ACI.

For ranking metric gains, sRank achieved 35.5% higher top-one accuracy than our DPR model with 7.5% less inference time. To evaluate robustness of sRank to template editing (the scenario where a physician further edits their templates after ranker training), we also verified that it achieved accuracy gains across subsets of physicians whose test encounters contain edited templates at various frequencies.

5 CONCLUSION

This paper presents our cross-attention learning-to-rank sRank model that we developed for two real-world industry tasks at Microsoft. We describe a number of optimizations and improvements that enable sRank to perform better on our tasks than previous models. The high quality and speed of sRank may make it an attractive option for additional industry ranking tasks that require selecting top-one options from candidate sets.

REFERENCES

- [1] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank using Gradient Descent. In *Proceedings of the 22nd International Conference on Machine Learning*. ACM, 89–96.
- [2] Christopher JC Burges. 2010. From Ranknet to Lambdarank to Lambdamart: an overview. *Microsoft Research Technical Report* 11, 23-581 (2010), 81.
- [3] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. 129–136.
- [4] ONNX Runtime developers. 2021. ONNX Runtime. <https://www.onnxruntime.ai>. Version: x.y.z.
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. <https://doi.org/10.18653/v1/N19-1423>
- [6] Seppo Enarvi, Marilisa Amoia, Miguel Del-Agua Teba, B. Delaney, Frank Diehl, Stefan Hahn, Kristina Harris, Liam R. McGrath, Yue Pan, Joel Pinto, Luca Rubini, Miguel Ruiz, Gagandeep Singh, Fabian Stemmer, Weiyi Sun, Paul Vozila, Thomas Lin, and Ranjani Ramamurthy. 2020. Generating Medical Reports from Patient-Doctor Conversations Using Sequence-to-Sequence Models. In *NLPMC*.
- [7] Luyu Gao, Zhuyun Dai, and Jamie Callan. 2020. Modularized transformer-based ranking framework. *arXiv preprint arXiv:2004.13313* (2020).
- [8] Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. 2020. Learning-to-Rank with BERT in TF-Ranking. *arXiv preprint arXiv:2004.08476* (2020).
- [9] Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*. Springer, 84–92.
- [10] 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 international conference on Information & Knowledge Management*. 2333–2338.
- [11] Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufman, Balint Miklos, Greg Corrado, Andrew Tomkins, Laszlo Lukacs, Marina Ganea, Peter Young, and Vivek Ramavajjala. 2016. Smart Reply: Automated Response Suggestion for Email. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD) (2016)*. <https://arxiv.org/pdf/1606.04870v1.pdf>
- [12] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906* (2020).
- [13] Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 39–48.
- [14] Tom Knoll, Francesco Moramarco, Alex Papadopoulos Korfiatis, Rachel D. Young, Claudia Ruffini, Mark Perera, Christian Perstl, Ehud Reiter, Anya Belz, and Aleksandar Savkov. 2022. User-Driven Research of Medical Note Generation Software. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 385–394.
- [15] Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2020. Parade: Passage representation aggregation for document reranking. *arXiv preprint arXiv:2008.09093* (2020).
- [16] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.
- [17] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonello, Nazli Goharian, and Ophir Frieder. 2020. Efficient document re-ranking for transformers by precomputing term representations. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 49–58.
- [18] George Michalopoulos, Kyle Williams, Gagandeep Singh, and Thomas Lin. 2022. MedicalSum: A Guided Clinical Abstractive Summarization Model for Generating Medical Reports from Patient-Doctor Conversations. In *Findings of the Association for Computational Linguistics: EMNLP 2022*.
- [19] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.
- [20] Soumava Kumar Roy, Mehrtash Harandi, Richard Nock, and Richard Hartley. 2019. Siamese networks: The tale of two manifolds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3046–3055.
- [21] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).
- [22] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web*. 373–374.
- [23] Michael Taylor, John Guiver, Stephen Robertson, and Tom Minka. 2008. Sofrank: Optimizing Non-smooth Rank Metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*. ACM, 77–86.
- [24] N Thakur, N Reimers, J Daxenberger, and I Gurevych. 2021. Augmented SBERT: data augmentation method for improving bi-encoders for pairwise sentence scoring tasks 2020. *arXiv preprint arXiv:2010.08240* (2021).
- [25] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. 2009. Learning to Rank by Optimizing NDCG Measure. In *Advances in Neural Information Processing Systems*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (Eds.), Vol. 22. Curran Associates, Inc. <https://proceedings.neurips.cc/paper/2009/file/b3967a0e938dc2a6340e258630febd5a-Paper.pdf>
- [26] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The Lambdaloss Framework for Ranking Metric Optimization. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1313–1322.
- [27] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. 2008. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*. 1192–1199.
- [28] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems* 33 (2020), 17283–17297.
- [29] Giulio Zhou and Jacob Devlin. 2021. Multi-Vector Attention Models for Deep Re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 5452–5456.
- [30] Xiaofeng Zhu and Diego Klabjan. 2020. Listwise learning to rank by exploring unique ratings. In *Proceedings of the 13th international conference on web search and data mining*. 798–806.