October 2021

Neural Approaches to Feedback in Information Retrieval

Keping Bi
University of Massachusetts Amherst

NEURAL APPROACHES TO FEEDBACK IN INFORMATION RETRIEVAL

A Dissertation Presented

by

KEPING BI

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2021

College of Information and Computer Sciences

NEURAL APPROACHES TO FEEDBACK IN INFORMATION RETRIEVAL

A Dissertation Presented

by

KEPING BI

Approved as to style and content by:

W. Bruce Croft, Chair

James Allan, Member

Daniel Sheldon, Member

Rajesh Bhatt, Member

James Allan, Chair
College of Information and Computer Sciences

To my parents

ACKNOWLEDGMENTS

I would like to express my gratitude to those who have helped me during my Ph.D. study and make the journey enjoyable and precious.

First and foremost, I would like to thank my adviser W. Bruce Croft for his guidance, consideration, and support. He gave me the perfect balance of guidance on the research direction and freedom to conduct the topics I am interested in. When I did not have a smooth research experience in my first two years, with his support, supervision, and encouragement, I survived the tough period and made essential progress afterward. His research vision and attitude have influenced me on understanding the importance of a research topic and how to conduct meaningful research. Without him, I could not have finished this dissertation and proceeded to the next stage of my career.

I am grateful to my committee members, James Allan, Daniel Sheldon, and Rajesh Bhatt. Their valuable comments and suggestions helped me refine the dissertation to be more clear and better presented. Especially, I thank James for the edits on my dissertation and all the brilliant questions during my defense. I thank Daniel for the beneficial discussions on the motivations of my model design and training objectives. Also, I thank Benjamin Marlin for his feedback and suggestions during my synthesis project, which finally becomes a part of the dissertation.

I would like to thank my mentors during internships, Choon Hui Teo, Rahul Jha, Asli Celikyilmaz, Pavel Metrikov, Chunyuan Li, and Byungki Byun. I thank Choonhui for supporting me in doing a research project that is related to my thesis topic but hard to be shipped online in the short term at Amazon. I thank Rahul

and Asli for giving me the freedom to do extractive summarization at Microsoft from my perspective developed from the experience in IR. I thank Pavel, Chunyuan, and Byungki for helping me understand user behaviors in email search better and providing insightful opinions during my internship at Microsoft.

I would like to thank the staff in our lab and the department. I appreciate the technical support from Dan Parker on the servers and equipments in the lab. Thank Kate Moruzzi, Jean Joyce, Glenn Stowell, and Stephen Harding for their dedicated administrative and logistical support. I thank our Graduate Programs Managers and Assistants, Eileen Hamel, Malaika Ross, and Leeanne M. Leclerc for their help at each milestone during my entire program.

I thank my current and previous labmates who have created a congenial atmosphere in our lab. In alphabet order: Ali, Chen, Dan, Hamed, Helia, Jiepu, John, Lakshmi, Liu, Myung-ha, Negin, Puxuan, Qingyao, Rab, Shahrzad, Sheikh, Shiri, Shiva, Weize, Yen-Chieh, Youngwoo, and Zhiqi. Discussions with them have provided a lot of insights and it was fun for us to share life experiences.

At last, I would like to thank my family and my fiance Qingyao for their immense love and support that warms me in the coldness and encourages me to persist when I was self-doubt. Thank Qingyao for his loving care and all our experience together. Thank my parents and sisters who are always there for me on my good and bad days. I am so lucky to have them in my life. I would not have my current accomplishment without them.

This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF IIS-1715095. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

ABSTRACT

NEURAL APPROACHES TO FEEDBACK IN INFORMATION RETRIEVAL

SEPTEMBER 2021

KEPING BI

B.E., NANKAI UNIVERSITY

M.S., PEKING UNIVERSTIY

M.S., UNIVERSITY OF MASSACHUSETTS, AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor W. Bruce Croft

Relevance feedback on search results indicates users' search intent and preferences. Extensive studies have shown that incorporating relevance feedback (RF) on the top k (usually 10) ranked results significantly improves the performance of re-ranking. However, most existing research on user feedback focuses on word-based retrieval models such as the vector space model (VSM) and language model (LM) for information retrieval (IR). Recently, neural retrieval models have shown their efficacy in capturing relevance matching in retrieval but little research has been conducted on neural approaches to feedback. This leads us to study different aspects of feedback with neural approaches in the dissertation.

RF techniques are seldom used in real search scenarios since they can require significant manual efforts to obtain explicit judgments for search results. However,

with mobile or voice-based intelligent assistants being more popular nowadays, user feedback of result quality could be collected potentially during their interactions with the assistants. Due to the limit of display space or voice bandwidth, these scenarios argue for retrieval on answer passages instead of documents and iterative feedback on one result at a time rather than feedback on a batch of results. To this end, we study iterative feedback versus top-k feedback with a focus on answer passages. Moreover, we study both positive and negative RF to refine the re-ranking performance. Although effective, positive feedback is not always available since relevant results may not be ranked at the top, especially for difficult queries. Also, in most cases, it is more beneficial to find the first relevant result compared with finding additional relevant results. Thus, incorporating only negative feedback to identify relevant results is an important research topic. However, it is much more challenging to find relevant results based on negative feedback than positive feedback since relevant results are usually similar while non-relevant results could vary considerably.

We focus on the tasks of text retrieval and product search to study the different aspects of incorporating feedback for ranking refinement with neural approaches. Our contributions are: (1) we show that iterative relevance feedback (IRF) is more effective than top-k RF on answer passages and we further improve IRF with neural approaches; (2) we propose an effective RF technique based on neural models for product search; (3) we study how to refine re-ranking with negative feedback for conversational product search; (4) we leverage negative feedback in user responses to ask clarifying questions in open-domain conversational search. Our research improves retrieval performance by incorporating feedback in interactive retrieval and approaches multi-turn conversational information-seeking tasks with a focus on positive and negative feedback.

TABLE OF CONTENTS

Page

ACKNOWLEDGMENTS . . . . . **v**

ABSTRACT . . . . . **vii**

LIST OF TABLES . . . . . **xiii**

LIST OF FIGURES . . . . . **xv**

CHAPTER

1. INTRODUCTION **1**

1.1 Positive Feedback 3

1.1.1 Iterative Relevance Feedback for Answer Passage Retrieval 5
1.1.2 An End-to-end Neural Relevance Feedback Model for Product Search 6

1.2 Negative Feedback 8

1.2.1 Conversational Product Search based on Negative Feedback 9
1.2.2 Asking Clarifying Questions Based on Negative Feedback in Information-Seeking Conversations 10

1.3 Contributions 12

2. RELATED WORK AND BACKGROUND **14**

2.1 Relevance Modeling in Information Retrieval 14

2.1.1 Conventional Retrieval Models 14
2.1.2 Neural Approaches to Information Retrieval 16

2.2 Feedback Modeling in Information Retrieval 19

2.2.1 Relevance Feedback 19

2.2.2 Negative Feedback . . . 22
2.2.3 Iterative Feedback . . . 23

2.3 Conversational Information Retrieval . . . 24

2.3.1 Conversational Search and Question Answering . . . 24
2.3.2 Asking Clarifying Questions . . . 26

2.4 Information Retrieval Applications . . . 28

2.4.1 Retrieval of Unstructured Text . . . 28
2.4.2 Retrieval of Structured Products . . . 30

3. ITERATIVE RELEVANCE FEEDBACK FOR ANSWER PASSAGE RETRIEVAL . . . 32

3.1 Introduction . . . 32
3.2 Word-based IRF Models . . . 34
3.3 Passage Embedding based IRF Models . . . 37

3.3.1 Passage Embeddings . . . 38
3.3.2 IRF with Passage-level Semantic Similarity . . . 39

3.4 Experiments of Word-based IRF . . . 41

3.4.1 Experimental Setup . . . 42
3.4.2 Results and Discussion . . . 44

3.5 Experiments of Passage Embedding Based IRF . . . 46

3.5.1 Experimental Setup . . . 48
3.5.2 Iterative Feedback with Embeddings . . . 49
3.5.3 Retrieval Given One Relevant Passage . . . 51
3.5.4 Parameter Sensitivity . . . 55

3.6 Summary . . . 55

4. AN END-TO-END NEURAL RELEVANCE FEEDBACK MODEL FOR PRODUCT SEARCH . . . 58

4.1 Introduction . . . 58
4.2 Context-aware Product Search . . . 60

4.2.1 Problem Formulation . . . 60
4.2.2 Context Dependency Models . . . 60
4.2.3 Context-aware Embedding Model . . . 62
4.2.4 Model Optimization . . . 66

4.3 Experimental Setup . . . 68

4.3.1 Datasets . . . 68
4.3.2 Evaluation Methodology . . . 69
4.3.3 Baselines . . . 70
4.3.4 Model Training . . . 72

4.4 Results and Discussion . . . 73

4.4.1 Overall Retrieval Performance . . . 73
4.4.2 Effect of Short-term Context . . . 76
4.4.3 Effect of Long-term Context . . . 78
4.4.4 Effect of Embedding Size . . . 80

4.5 Summary . . . 80

5. CONVERSATIONAL PRODUCT SEARCH BASED ON NEGATIVE FEEDBACK . . . 81

5.1 Introduction . . . 81
5.2 Conversation Paradigm and Problem Formulation . . . 82
5.3 Aspect-value Likelihood Embedding Model . . . 84

5.3.1 Item Generation Model . . . 84
5.3.2 Query Representation . . . 85
5.3.3 User/Item Language Model . . . 85
5.3.4 Aspect-Value Generation Model . . . 86
5.3.5 Unified AVLEM Framework . . . 90
5.3.6 Item Ranking with AVLEM . . . 93

5.4 Experimental Setup . . . 94

5.4.1 Datasets . . . 94
5.4.2 Evaluation Methodology . . . 96
5.4.3 Baselines . . . 97
5.4.4 Model Parameter Settings . . . 99

5.5 Results and Discussion . . . 100

5.5.1 Overall Retrieval Performance . . . 100
5.5.2 Ablation Study . . . 104
5.5.3 Parameter Sensitivity . . . 105

5.6 Summary . . . 106

6. ASKING CLARIFYING QUESTIONS BASED ON NEGATIVE FEEDBACK IN INFORMATION-SEEKING CONVERSATIONS 107

6.1 Introduction 107
6.2 Conversation Intent Clarification 108

6.2.1 Task Formulation 109
6.2.2 First Clarifying Question Selection 110
6.2.3 Clarifying Intents Using Negative Feedback 112

6.3 Experimental Setup 115

6.3.1 Data 115
6.3.2 Evaluation 117
6.3.3 Baselines 118
6.3.4 Technical Details 120

6.4 Results and Discussion 121

6.4.1 Clarifying Question Selection Results 121
6.4.2 Document Retrieval Performance 125
6.4.3 Case Analysis 126

6.5 Summary 130

7. CONCLUSIONS AND FUTURE WORK 131

7.1 Overview of Neural Feedback Approaches in IR 131
7.2 Summary of Experimental Results 133
7.3 Future Work 135

BIBLIOGRAPHY 139

LIST OF TABLES

Table **Page**

3.1 Statistics of experimental datasets. 43

3.2 Performance of QL and BM25. 46

3.3 Performance of iterative feedback on document and answer passage collections. All the methods with feedback are significantly better than initial retrieval model (Initial). The initial ranking model is QL for RM3, Distillation, and BM25 for Rocchio. '*' denote significant improvements over the standard top-10 feedback model (10×1) in Fish's randomization test [123] ($p < 0.05$). 47

3.4 Performance of our proposed method with different paragraph representations compared with word-based and embedding-based RF baselines. '$*$' and '†' denote significant improvements over word-based (RM3, Distillation) and embedding-based (ERM) baselines respectively based on Fisher's randomization test [123] ($p < 0.05$). 50

3.5 Performance of different IRF methods on finding other relevant answers given one relevant answer. '$*$' and '†' denote significant improvements over word-based (RM3, Distillation, Rocchio) or embedding-based (ERM) baselines respectively in Fisher's randomization test [123] ($p < 0.05$). 53

4.1 Statistics of our collected datasets 68

4.2 Comparison of baselines and our short-term context embedding model (SCEM) on re-ranking when users paginate to the 2nd and 3rd page. The number is the relative improvement of each method compared with the production model (PROD)[1]. '$^-$' indicates significantly worse of each baseline compared with SCEM in student t-test with $p \leq 0.001$. Differences larger than 3% are approximately significant. 74

5.1 Statistics of Amazon datasets. 94

5.2 Examples of extracted aspect-value pairs. 96

5.3 Comparison between baselines and our model AVLEM. Numbers marked with'*' are the best baseline performance. '+' indicates significant differences between the iterative feedback models and their corresponding initial rankers in Fisher random test [123] with $p < 0.05$, i.e., Rocchio vs BM25, SingleNeg and MultiNeg vs QL, AVLEM$_{pos}$, AVLEM$_{neg}$ and AVLEM$_{all}$ vs AVLEM$_{init}$. '†' denotes significant improvements upon the best baseline. The highest value in each column is in bold. 101

6.1 Statistics of our revised version of Qulac. 117

6.2 Model performance on intent clarification task evaluated using only label 2 or both label 1 & 2. '*' indicates the best baseline results, and '†' shows the statistically significant improvements over them. 121

6.3 Document retrieval performance with conversations composed by each model. The best baseline results are marked with '*', and the statistically significant improvements over them are marked with'†'. 126

6.4 Good and bad cases of MMR-BERT compared with the best baseline - BERT-GT in terms of their MRR differences(ΔMRR(CQ)) in the intent clarification task. The maximal number of conversation turns is 5. ΔMRR(Doc) denotes the MRR difference of the associated document retrieval task after the conversation. Queries are shown in the format of *query(facet description); topic type; facet type.* 128

LIST OF FIGURES

Figure **Page**

3.1 An example conversation between a user and an assistant. 33

3.2 HDC models used in our experiments. Red words are local context, and blue words are global context. 40

3.3 Performance of our method with different paragraph representations compared with Rocchio. '+' means significant difference based on Fisher's randomization test [123] ($p < 0.05$) 51

3.4 Parameter sensitivity of the coefficient λ_{sf} in Equation (3.9) with PVC on iterative RF and retrieval given one relevant passage. 56

4.1 Different assumptions to model different factors as context for purchase prediction. 61

4.2 The structure of our context-aware embedding model (CEM). w represents words in queries or product titles; $\mathcal{C}_{1:t}$ denotes the click item set in the first t SERPs, which consist of item c; $\mathcal{S}_t$ is the overall context of the first t SEPRs, a combination of query q, user u and clicks $\mathcal{C}_{1:t}$; i is an item in the candidate set $\mathcal{D}_{t+1}$ for re-ranking from page $t+1$. 63

4.3 The effect of λ_c, λ_u, embedding size on the performance of each model in the collection of *Toys & Games* when re-ranking from the second SERP for the scenarios where users paginate to page 2. 77

5.1 A workflow of conversational search system based on negative feedback. 83

5.2 The architecture of our aspect-value likelihood embedding model (AVLEM). The solid and dotted arrows represent the generation from a multinomial and a multivariate Bernoulli distribution respectively. The shaded and blank background represents the occurrence and nonoccurrence of the target. v^+ and v^- denote positive and negative values. 86

5.3 The MRR of AVLEM with different components removed and parameter sensitivity analysis of baselines and AVLEM on *Cell Phones&Accessories*. 103

6.1 A workflow of the intent clarification task. 109

6.2 Our Maximal Marginal Relevance based BERT Model (MMR-BERT). 110

6.3 Comparison of MMR-BERT and baselines in terms of the cumulative number of success conversations at each turn on the intent clarification task. 122

6.5 MRR at each turn on document retrieval. 127

CHAPTER 1

INTRODUCTION

The essential target of an Information Retrieval (IR) system is to find the pieces of information that can satisfy a user's information need. Users may not express their search intent precisely with queries that usually consist of several words, especially when they have complex information needs. Their feedback on retrieved results is a strong signal to indicate their search intent and helps IR systems differentiate relevant information from non-relevant ones. Extensive studies have been conducted on extracting beneficial terms from the top k results with relevance judgments to expand the initial query and re-ranking based on the new query model [117, 119, 166, 127, 20, 43]. These relevance feedback (RF) techniques are shown to be much more effective compared with retrieval based on initial queries.

Most existing feedback approaches, however, are based on conventional retrieval models built on top of bag-of-words representations, such as the vector space model [120] and language model (LM) for information retrieval (IR) [103], which can sometimes fail to capture semantic meanings during relevance matching. In recent years, neural models and embeddings have shown their superiority in representing and matching semantics and have achieved state-of-the-art performance in many natural language processing (NLP) and retrieval tasks [93, 80, 125, 24, 44]. Words with similar meanings have similar dense representations in the semantic space. Afterward, research efforts on modeling relevance matching have moved from feature engineering to neural architecture designing. There has been significant progress in building effective neural models for document and answer retrieval [65, 54, 147, 98]. However,

little research focuses on neural approaches that incorporate feedback information. In this dissertation, we study how to effectively incorporate user feedback with neural approaches to facilitate various information retrieval tasks.

Despite their effectiveness, feedback techniques are seldom used based on relevance assessments in real search scenarios since they can require significant manual efforts to obtain explicit judgments from users on their search results. Also, research in this direction has become scarce in recent years. However, it is still feasible to collect user feedback in some scenarios. User behaviors such as clicks and skips on the search results pages (SERPs) can be collected easily. They indicate the result quality and can be considered as implicit feedback [69, 165, 172]. In addition, as intelligent assistants such as Siri and Cortana become more popular, user feedback about result quality could potentially be obtained during users' interactions with the assistants. A retrieval system can ask for user feedback on not only the quality of retrieved documents or passages but also clarifying questions and the potential reasons for an unfavorable result. Following reasonable conversation paradigms, the system can collect various types of user feedback naturally. The display space or voice bandwidth in such a system leads to severe limitations on the length and number of results shown in a single interaction. Thus, these scenarios argue for collecting feedback on text of short length and iteratively with one result at a time.

There are mainly two types of techniques incorporating user feedback, which are positive feedback and negative feedback. Most existing research focuses on relevance feedback where a topic model is constructed based on results with positive feedback to retrieve more relevant results. Non-relevant results can also be used in relevance feedback techniques to distill the relevant topic model [117, 20]. When both positive and negative feedback are available, judged relevant results play a more important role than non-relevant results in identifying other relevant results [1]. Negative feedback approaches usually build models with only non-relevant results to retrieve the first

relevant results. Relevant results are usually similar while the reasons for a result to be non-relevant can be varied. Research on negative feedback has been sparse [139, 140, 71] compared to positive feedback since it is more challenging to find relevant results with non-relevant ones than using known relevant ones.

In this dissertation, we study different aspects of incorporating user feedback for ranking refinement in various interactive IR scenarios. We start with how to effectively incorporate positive feedback. We compare iterative feedback with top-k feedback (one-shot reranking based on top k results) and investigate how to improve iterative ranking performance for various retrieval models. Then we dig into the more challenging direction of how to use negative feedback to identify user intent and retrieve satisfactory results. In the following sections, we introduce various aspects of incorporating feedback in different IR scenarios and give an overview of our contributions.

1.1 Positive Feedback

Positive feedback aims to find more relevant results given some known relevant results confirmed by users. It is beneficial in the cases when one relevant result is not enough to satisfy user needs such as a user searches for existing literature on a research topic or applicable laws to a certain scenario, etc. These tasks emphasize the recall of retrieved results and try to find more information related to the target topic. When a topic is ambiguous or has multiple subtopics, users' positive feedback on results regarding a certain meaning or subtopic indicates their search intent. A retrieval system could be able to tailor the results shown to the users next based on their positive feedback.

Non-factoid question answering and product search are two IR tasks where positive feedback is potentially beneficial. In contrast to factoid questions that can be answered with a number or entity and a single correct answer can satisfy the infor-

mation need, non-factoid questions need answer passages that cover details about the topic and multiple relevant answers are usually preferable to provide more information. For example, for the question "What are the methods to control type ii diabetes?", multiple answers from different sources or that cover varied methods could be beneficial. In product search, user purchases depend on not only product relevance but also user preferences. A user may like items that have certain characteristics and he/she purchases one from them finally. Take the query "women swimsuits" for example, the user likes swimsuits that are one piece with boy legs and could browse several such types of items before making an order. User clicks during browsing indicate their preferences and can be considered as implicit positive feedback. Showing more items that match user preferences according to their positive feedback could be helpful.

Existing studies on relevance feedback (RF) mainly focus on using positive feedback on top k documents and have achieved compelling performance in adhoc document retrieval [117, 114, 119, 1, 68, 12, 118, 79, 166, 43, 20]. However, how they would perform in new scenarios such as mobile search and product search is unknown. To adapt to the scenarios where user feedback can be potentially obtained, we explore iterative feedback on answer passages and implicit feedback on products.

Previous RF methods are mainly built upon word-based retrieval models since they were proposed before neural models have become a prevailing way to approach a variety of NLP and IR tasks. As a first step, we investigate how to build effective RF models based on conventional retrieval models leveraging the advantages of neural embeddings in an unsupervised way. Then we study how to refine neural ranking models with the positive feedback information by learning a supervised model. In the following subsections, we introduce the tasks of answer passage retrieval and multi-page product search for which we study iterative feedback versus top-k feedback and

propose neural feedback models on top of word-based retrieval models and neural retrieval models.

1.1.1 Iterative Relevance Feedback for Answer Passage Retrieval

Most RF techniques are originally designed for document retrieval and they aim to build a more informative query model by introducing expansion terms and re-weighting the original query terms based on the documents with relevance judgments. It is essential to have sufficient text for the accurate estimation of word weights in the RF models. In iterative RF, the information available to extract expansion terms is much less compared to standard top-k feedback, which may harm the model accuracy. Moreover, they contain even less content for existing RF methods to estimate word weights accurately. However, on the one hand, answer passages are usually more focused than documents, which may reduce noise during model estimation. Thus, it is worth investigating whether iterative feedback is effective for document and answer passage retrieval.

To counteract the limitation of less text in iterative answer retrieval and further improve the performance of IRF in answer passage retrieval, we introduce complementary information from semantic space to help estimate a more accurate RF model. Dense vector representations of words and paragraphs in distributed semantic space, called embeddings, [93, 80, 125, 40, 24], have been effectively applied on many natural language processing (NLP) tasks, such as sentiment analysis and word analogy. Embeddings have been used in some previous research on pseudo relevance feedback based on documents [161, 112], but their impact in iterative and passage-based feedback is not known. Moreover, they only use semantic similarity at the term level and do not consider semantic matching of larger granularity. Paragraph vectors [80, 125, 24], specially designed to represent the topic of a paragraph in the semantic space as a whole, offer a new way to measure the similarity between answer passages in terms of

larger granularity. This has led us to focus on designing techniques of IRF for answer passage retrieval with paragraph embeddings to improve upon word-based IRF and other embedding-based IRF using term-level semantic similarity.

In this dissertation, we investigate whether iterative feedback based on different frameworks is effective relative to RF with a list of top k (k=10) results on both document and answer passage retrieval. To further improve the performance of iterative RF on answer passages, we leverage paragraph vectors to capture passage-level semantics. We study whether passage-level semantic matching is beneficial to word-based RF models, whether it is more effective than term-level semantic matching, and whether the semantic information from both granularities complements each other.

1.1.2 An End-to-end Neural Relevance Feedback Model for Product Search

In product search, users need to pay money for their purchased products, which costs much more than clicking on links in Web search. Thus, in contrast to Web search where people check results on the first search result page (SERP) most of the time, users are more likely to be patient and browse items in multiple SERPs before deciding which one to purchase. Their clicks in previous SERPs can reflect their preferences, based on which the system can refine the results in subsequent SERPs accordingly. From the analysis of Amazon search log [19], we observe that in about 5% to 15% of the search traffic, users browse and click results in the previous pages and purchase items in the later result pages. Thus, we study how to incorporate positive feedback in the task of multi-page product search.

Existing RF techniques can also be used in multi-page product search if we consider user clicks as positive feedback on items. These methods, however, are designed specifically for text retrieval, which could not be effective for products. Product search has significant differences from text retrieval. Texts are unstructured data while products are structured data that has many aspects such as brand, color, size,

etc. In product search, relevance is not enough to measure result quality as the goal is to elicit user purchases which also depend on user preferences. Moreover, as conventional retrieval models, existing RF methods are mostly unsupervised without the need for labeled data for training. Since each query requires annotations of a large number of documents, most document retrieval collections have only a few hundreds of queries. The unsupervised RF models are competent on these collections. Contrariwise, in product search, user purchases can be obtained from search logs automatically without the need for manual annotation. The sufficient amount of ground-truth data argues for a more effective supervised model guided by user purchases for the task. With this regard, we propose to learn a supervised end-to-end neural relevance feedback model based on a neural retrieval model for product search.

In this dissertation, we leverage user clicks as implicit positive feedback to provide users with more tailored results in product search. Concretely, we reformulate product search as a dynamic ranking problem, i.e., when users request the next SERPs, the remaining unseen results will be re-ranked based on the user clicks in the previous SERPs. We introduce several context dependency assumptions for the task to capture users' short-term and long-term, and long-short-term preferences. User feedback within a query session is considered as short-term context and user identifiers across all the query sessions are the long-term context. We propose an end-to-end context-aware neural embedding model that can represent each assumption by changing the coefficients to combine long-term and short-term context. We investigate whether user feedback is beneficial to the retrieval model without using feedback, how user feedback as short-term context performs compared to long-term context, and whether our end-to-end neural feedback model is better than unsupervised RF methods.

1.2 Negative Feedback

Research on negative feedback focuses on identifying relevant results based on the non-relevant results collected from user feedback. The target is to promote the first relevant result to higher positions so that users can see them within fewer iterations. An effective negative feedback technique would bring more benefit to a retrieval system than positive feedback since it is more valuable to show the first relevant result to users compared to showing additional relevant results to users. In many cases, one relevant result is enough to satisfy a user need, such as when the user wants to know the definition of a slang word or the eligibility to get a COVID-19 stimulus check. Moreover, negative feedback is more often available than positive feedback since non-relevant results are much more common than relevant ones, especially when the search query is too difficult for the system to retrieve relevant results at the top.

Despite the value of negative feedback, there have been many fewer existing studies on it compared to positive feedback [139, 140, 71]. This is probably due to the greater challenge to effectively leverage negative feedback for ranking refinement. Relevant results often share common characteristics while the reasons for a result to be non-relevant could be varied. Existing negative feedback approaches were originally proposed to improve the performance of different queries in document retrieval, for which none of the top k documents are relevant. They typically extract negative topic models from the non-relevant documents and give lower scores to the results that are similar to the negative topic models during re-ranking. However, such methods have limited gains due to the large amount of non-relevant results and the lack of ability in measuring semantic similarity using word-based retrieval models. In conversational search systems that prefer short texts and showing one result at a time to users, results with negative feedback have many fewer words for negative topic model estimation, which makes it even harder for existing negative feedback methods to effectively filter out similar results.

To improve upon the previous negative feedback methods, we explore two ways of obtaining feedback in different scenarios and propose neural models that can incorporate the feedback accordingly. First, when users provide negative feedback to a result, we further ask users about the fine-grained feedback on the detailed aspects of the result and use the detailed feedback instead of the result-level feedback for re-ranking refinement. Since products are naturally structured and have aspects such as material, color, brand, etc., and associated values, we study breaking result-level negative feedback to aspect-value-level feedback in conversational product search. Second, instead of collecting feedback on retrieved documents, we explore asking clarifying questions about an aspect or subtopic of user queries for feedback before showing retrieval results. Based on the negative feedback to the clarifying questions, the system aims to ask questions on other aspects. The intent space under a query is much smaller than the space of documents and we can leverage contextual neural language models pretrained with large-scale data such as BERT [44] to better measure semantic similarity. Thus, it can be promising to ask clarifying questions based on negative feedback and we study this problem on open-domain conversational search.

In the following subsections, we introduce the tasks for which we study negative feedback, i.e., conversational product search based on negative feedback, and asking clarifying questions based on negative feedback in information-seeking conversations.

1.2.1 Conversational Product Search based on Negative Feedback

In product search, when users do not like an item, they may still favor some aspects of the item although they are not satisfied with the item's certain attributes for sure. If we only consider negative feedback on the item level, potential ideal items may be filtered out as well when they share some common characteristics with the non-satisfactory item. Since products are structured and user preferences are based on their aspects and associated values, a conversational system can ask for

users' detailed preferences on the item's aspect-values after they provide negative feedback to an item. For instance, when a user says he/she does not like a presented mobile phone, it is easy to gather their feedback on the aspect-value pairs such as "brand-Huawei", "screen-curved", and "battery-removable". In this way, the result-level negative feedback can be further decomposed to aspect-value-level positive and negative feedback, which could be much more indicative of user preferences.

As existing relevance or negative feedback techniques are designed for text data, it argues for a novel model architecture to leverage the aspect-value-level feedback on the products. The feedback model should be able to incorporate both the fine-grained positive and negative feedback with the ranking model effectively. It is challenging to do so since the semantics of product aspects and values need to be captured, such as "battery-replaceable" versus "battery-removable", and "appearance-stylish" versus "style-fashionable". Negative feedback is especially difficult to use for its opposite effect in matching an item compared to positive feedback. In Chapter 5, we study how to build an effective feedback model based on a state-of-the-art product search method and compare its performance with methods without using feedback and only using item-level negative feedback.

1.2.2 Asking Clarifying Questions Based on Negative Feedback in Information-Seeking Conversations

Another way of collecting user feedback for ranking refinement is to ask questions about the meanings or subtopics of a query to clarify user intent before showing retrieved results. In an open-domain information-seeking scenario, for example, given a user query "rice", the system can ask "Are you looking for information about Rice University?", "Are you looking for the types of rice?", or "Do you want to know recipes about rice?". When the user provides negative feedback, the next question to ask should have a different meaning or cover other subtopics from the previous one. The

negative feedback to such clarifying questions usually indicates an information need about certain meanings or subtopics. This is more helpful to demote non-relevant information than negative feedback to a passage or document that can be too specific to filter out general information. The intent space under a query is much smaller than the space of documents, which makes it possible for negative feedback to be effective in identifying user intent.

For the target of asking a question about different intents from previously asked questions, it is essential to measure the semantic similarity between questions. Using question words alone for measuring similarity cannot take semantic matching into account and is not accurate. Similar to what we do in iterative RF, we again need to leverage neural models to capture the semantics of the questions. Pretrained contextual language models such as BERT [44], XLNet [157], and GTP3 [21] have shown good abilities to understand language. They are pretrained with a large scale of documents such as Wikipedia Webpages so that they see enough texts of diverse topics and various contexts of each word. The word representations in such models depend on their contexts and are dynamic, different from the static word embeddings in models like Word2Vec [93] and Glove [101]. By fine-tuning the models with local corpora, they have achieved state-of-the-art performance in many NLP tasks. To use negative feedback effectively to ask clarifying questions, we also leverage the power of these advanced embeddings. In Chapter 6, we introduce how to build an effective negative feedback model based on BERT to select clarifying questions, how the model performs compared to conventional negative feedback methods as well as state-of-the-art baselines that are also based on BERT but do not process negative feedback in user responses specifically, and how the asked questions impact the performance of the associated document retrieval task.

1.3 Contributions

The contributions of this dissertation are as follows:

- We convert standard top-k RF techniques to iterative models and show that iterative feedback is at least as effective as standard methods for document retrieval and more effective for answer passage retrieval. To effectively find more relevant passages with users' explicit positive feedback, we propose an unsupervised passage embedding-based IRF method rather than the term level which most previous work focuses on. We show that passage-level semantic matching can produce significant improvements compared to different word-based RF methods and is more effective than term-level semantic matching in IRF. The semantic match information from both granularities are complementary to each other and lead to even better performance together (Chapter 3).

- We reformulate conventional one-shot ranking in product search to dynamic ranking, i.e., multi-page search, based on user clicks, to study leveraging implicit positive feedback to find users' ideal items. We introduce different context dependency assumptions for a product search query session and the short-term context dependency indicates user clicks in a query session before the user requests for another search result page. We propose a simple yet effective end-to-end embedding model to capture different types of dependency. The model can act as both a retrieval model without using feedback and a feedback model that can incorporate user clicks for finding ideal items. Based on the data collected from real search logs, our experimental results confirm the effectiveness of the supervised end-to-end feedback model in incorporating user clicks as short-term context. It significantly outperforms competitive word-based feedback models and the neural retrieval models that either use only long-term context or no context (Chapter 4).

- We propose a paradigm for conversational product search based on negative feedback, where the system identifies users' preferences by showing results and collecting feedback on the aspect-value pairs of the non-relevant items. To incorporate the fine-grained feedback, we propose an aspect-value likelihood model that can cope with both positive and negative feedback on the aspect-value pairs. It consists of the aspect generation model given items, and the value generation model given items and aspects. One multivariate Bernoulli (MB) distribution is assumed for the aspect generation model, and two other MB distributions are assumed for the generation of positive and negative values. Experimental results show that our model significantly outperforms the state-of-the-art product search baselines without using feedback and baselines using item-level negative feedback (Chapter 5).

- We propose an intent clarification task grounded on yes/no questions where the objective is to select the correct questions covering user intents within the fewest conversation turns using negative feedback. We leverage the pretrained contextual language models - BERT to help differentiate semantic meanings of questions and propose a maximum-marginal-relevance-based BERT model (MMR-BERT) to incorporate negative feedback in the conversation history for clarifying question selection. We show that MMR-BERT significantly outperforms state-of-the-art baselines and conventional negative feedback methods in both the intent clarification task and the associated document retrieval task (Chapter 6).

CHAPTER 2

RELATED WORK AND BACKGROUND

There are four strands of research topics related to the dissertation. They are relevance modeling, feedback modeling, conversational information retrieval, and the IR applications we study feedback approaches on.

2.1 Relevance Modeling in Information Retrieval

Retrieval models are the foundation of IR systems and the basis for feedback models to incorporate feedback information. Relevance is the most important criterion for retrieval models to score results. Their history dates back to the mid-1900s. Conventional retrieval models are usually based on word matching. In recent years, deep learning techniques have shown compelling performance in various domains, inspiring the IR community to explore neural approaches for retrieval models. We introduce some related studies on conventional word-based and recent neural retrieval models in this section.

2.1.1 Conventional Retrieval Models

There are mainly three types of retrieval models: the Vector Space Model (VSM) [120], the probabilistic model[88, 113], and the Language Modeling (LM) approach for IR [103].

Vector Space Model. The vector space model [120] generally assumes that each word represents a dimension in the space and documents or queries are points in the space represented with vectors. A typical weighting scheme to determine the

value for each dimension is the term frequency-inverse document frequency (TF-IDF). The relevance between a document and a query can be measured with the similarity between the two vectors, such as their cosine similarity.

Probabilistic Model. The probabilistic models rank documents according to their posterior probability of relevance [88, 113, 116]. They assume that relevance is a basic, dichotomous, criterion variable independent of IR systems. Let R be the binary random variable for relevance, then the probability of a document d being relevant to a query q can be represented by $P(R|q,d)$. By using the Bayes rule, we have $P(R|q,d) = P(d|R,q)P(R|q)/P(d|q)$. A typical way to rank documents is using the ratio of $P(R|q,d)/P(\overline{R}|q,d)$ [88, 113], where $\overline{R}$ denotes irrelevance. Thus, it is essential to compute $P(d|R,q)$ to produce ranking scores. A well-known model in the probabilistic framework is the Binary Independent Retrieval model (BIR) [158], which assumes that documents are binary vectors and term occurrences are conditionally independent in the set of relevant or irrelevant documents. The relevance information of the query is not known in advance, so numerous techniques have used the statistics about the collection of documents to estimate the $P(t|R,q)$ and $P(t|\overline{R},q)$, where t is a term in d [63, 158, 115, 116]. Among them, BM25 [116] is the most popular, which assumes that term frequencies follow the mixture of two Poisson distributions. Since probabilistic models also use bag-of-words representations, they can be treated as an instance of VSM. Instead of ranking with $P(t|R,q)$ and $P(t|\overline{R},q)$, some studies use probabilistic assumptions to estimate term weights and similarity functions such as dot product between the query and document vectors for ranking [158, 114, 116].

Language Modeling Approach for IR. Ponte and Croft [103] proposed the Language Modeling (LM) approach for IR (LMIR) in the late 1990s, which motivated a new direction of research in this framework. The LM approach does not model relevance separately as the probabilistic models do. Instead, it ranks a document d according to the probability it is generated from the query q, i.e., $P(d|q)$, which can be

computed by $P(d|q) = P(q|d)P(d)/P(q)$. Since $P(q)$ is the same for all the documents and the prior distribution of $P(d)$ is often treated as uniform, we can simply rank results according to $P(q|d)$. The query is assumed to be generated based on a set of unigrams from a multinomial distribution, and $P(q|d)$ is computed based on maximum likelihood estimation. The original LMIR cannot handle the case when query words do not occur in documents. Noticing the problem, Zhai and Lafferty [167] have studied various smoothing methods to account for the absent query words. Some work has extended the unigram assumption in the original LM approach for IR to bigrams and trigrams [52, 91, 15]. For instance, Gao et al. [52] built a dependence language model that computes the probability of generating queries based on bigrams. Metzler and Croft [91] proposed to combine the language models of unigrams and bigrams in ordered and unordered windows with Markov random field (MRF). Bendersky and Croft [15] extended the MRF model by finding and dynamically weighting concepts in queries. These methods have achieved compelling performance in ad-hoc retrieval by only considering exact term matching.

In Chapter 3, we convert various RF techniques in the VSM and LMIR framework to iterative versions and conduct extensive comparisons between our proposed IRF models and baselines in both VSM and LMIR framework.

2.1.2 Neural Approaches to Information Retrieval

Embedding-based Approaches. Initial attempts on using deep learning techniques for IR focused on incorporating embeddings of words or passages to conventional models so that semantic matching can also be captured during relevance modeling. Word embeddings, also referred to as distributed representations of words, are dense vectors that are mapped from the vocabulary space V to a d-dimensional latent space, where d is much smaller than $|V|$. Words with similar meanings are mapped to nearby points in the embedding space. Word2Vec [93, 92] and Glove [101]

are two well-known word embedding techniques. They have similar ideas of learning word embeddings by predicting its context words or using the embeddings of its adjacent words to predict itself. However, Word2Vec [93] is based on a feedforward neural network while Glove [101] decomposes a global word co-occurrence matrix.

To also measure semantic match during relevance modeling, similarities of word embeddings can be used to compute the transition probabilities between words [51, 112, 162, 161, 55] and incorporated with the conventional retrieval models. For instance, Ganguly et al. [51] refined the original language modeling approach for IR (LMIR) [103] by smoothing document language models using the semantic similarities between terms in the vocabulary. Zheng and Callan [173] learned a supervised model to estimate term weights with distributed word vectors and refine the original LMIR [103] or BM25 [116].

Paragraph Vectors (PV) [80, 40, 125, 24] generalize the idea of Word2Vec [93] to learn the representation for documents, i.e. a target word is predicted using the document alone or together with the context words. Ai et al. [6, 5] use the generative probability of a word given the paragraph vector of a document to smooth the original word-based language model in ad-hoc retrieval. Our studies on iterative RF for answer passages in Chapter 3 also leverage paragraph vectors to obtain passage-level semantic matches.

Neural Ranking Models. Besides incorporating embeddings to conventional retrieval models, there have been studies on neural ranking models that are trained end-to-end with relevance labels as guidance. The emerging of such models has switched the efforts of feature engineering to the careful design of neural architectures. Initially, the models focused on representing queries and documents and calculated the ranking score with their cosine similarity. DSSM [65] that uses feed-forward networks and C-DSSM [122] that refine the representations with convolutional neural networks are two examples. However, such models are usually effective on short texts and

have much less compelling performance on long documents. Guo et al. [54] pointed out the necessity of building models that focus on interactions between query and document terms instead of learning embedding vectors of queries and documents and matching their similarities. Their insight has shed light on the following work in this direction. Lu and Li [84] proposed DeepMatch that considers localness and hierarchy when modeling the interactions. Xiong et al. [147] applied various kernel functions to the query-document term interaction matrix in a model called KNRM. Dai et al. [42] further extended KNRM by capturing soft matching of n-grams with convolutional neural networks and named the updated model as Conv-KNRM. Pang et al. [98] proposed DeepRank that divides documents into passages with a sliding window and aggregates the signals of all the query-passage matching to model relevance. Fan et al. [50] introduced a hierarchical neural matching model that learns passage-level relevance signals and makes the global decision based on both individual and accumulative passage relevance. More information about neural ranking models can be found in the comprehensive survey paper [56].

More recently, contextual neural language models that represent words dynamically according to their context have achieved compelling performance on a wide range of natural language processing tasks (NLP), such as ELMo [102], BERT [44], XLNet [157], and GPT3 [21]. ELMo [102] is based on recurrent neural networks (RNNs) while most other models [44, 157, 21] are based on the transformer [134] architecture. With a tremendous number of parameters pretrained with enormous data, they have shown good abilities to understand language. Inspired by this, several studies have explored leveraging BERT in IR tasks as well. Nogueira and Cho [96] have shown BERT's effectiveness on passage ranking. Dai and Callan [41] demonstrated that BERT can leverage language structures better and enhance retrieval performance on queries in natural languages. Wu et al. [144] proposed a passage cumulative gain model that applies a sequential encoding layer on top of the BERT output of a query-passage

pair to score a document. We also employ BERT to facilitate our studies on asking clarifying questions based on negative feedback in Chapter 6.

2.2 Feedback Modeling in Information Retrieval

We review the related work on feedback modeling from three aspects: relevance feedback (RF), negative feedback, and iterative feedback.

2.2.1 Relevance Feedback

In general, previous research has used three main types of relevance feedback methods for ad-hoc retrieval [1], which are based on the VSM [120], the probabilistic model [88], and the LMIR [103]. Most of them extract expansion terms from annotated relevant documents and re-weight the original query terms to estimate a more accurate query model to retrieve better results.

Rocchio [117] is generally credited as the first RF technique, developed using the VSM. It refines the vector of a user query by bringing it closer to the average vector of relevant documents and further from the average vector of non-relevant documents. Ide [67] extended Rocchio by only considering top non-relevant documents for feedback. These techniques originally used all terms in relevant documents for query expansion. However, Harman [60] showed that it was more effective to use only selected terms from relevant documents than all the terms. Robertson and Jones [114] proposed the first method based on the distribution of query terms in relevant and non-relevant documents. This method only reweighs query terms without adding any expansion terms. Harper et al. [62] proposed to expand a query by adding all the terms directly connected to each query term in a maximum spanning tree built with term-term clustering. Harman [61] confirmed the importance of query expansion in

[1]Due to the efforts required in true RF, numerous studies have switched to pseudo RF where top k results are assumed to be relevant. Some of these pseudo RF methods can also be applied to true RF, so we also introduce some of the pseudo RF models that are applicable to true RF.

addition to term reweighting for a probabilistic RF model. Cox et al. [35] proposed an RF method for image retrieval using the Bayes rule to predict the probability distribution over possible image targets rather than refining query. Vinay et al. [135] investigated the performance of VSM-based and probilistic RF models proposed in [117], [114], and [35] on Web search. They showed significantly various upper bound performance of the three techniques. Salton et al. [119] studied and compared various RF techniques based on the VSM and probabilistic model on six collections. Many other variations are also introduced in [118], which includes a comprehensive survey of feedback approaches in IR.

More recently, feedback techniques have been investigated extensively based on the LM approach, among which the relevance model [79] and the mixture model [166] are two well-known examples that empirically perform well. In a typical version of the relevance model (RM3) [79], the probabilities of expansion terms are estimated with occurrences of the terms in feedback documents. The mixture model [166] considers a feedback document to be generated from a mixture of a corpus language model and a query topic model, which is estimated with the EM algorithm. Some recent work [85, 20, 43] extends the mixture model by considering additional or different language models as components of the mixture. Specifically, Lv et al. [85] used relevant, non-relevant, and unlabelled documents to estimate the relevance model. Brondwine et al. [20] proposed a distillation model that distills the relevance topic model from the mixture using non-relevant documents. Dehghani et al. [43] considered each feedback document is generated from the mixture of significant words model, general model, and specific model.

Other aspects of relevance feedback have also been studied. Allan [11] claimed that RF performs better with the best-matched passages in relevant documents than with full documents. Lv and Zhai [86] proposed learning an adaptive coefficient for combining the original query and feedback model instead of using a fixed value. Diaz

[46] re-ranked documents with score regularization, i.e., forcing scores of documents related to relevant documents to be high and those related to non-relevant documents to be low. Tan et al. [127] proposed to collect feedback directly on terms instead of documents. Can et al. [22] incorporated relevance feedback to do query-specific modification in a learning-to-rank model. Rabinovich et al. [107] studied RF on document lists which comprise several intermediate retrieved lists fused. Vassilvitskii and Brill [133] used the web-graph distance between documents to rerank results rather than using query refinement in Web search.

Neural Approaches to RF. Initial exploration of neural approaches to RF can be found in the 1990s. Crestani [36] used a 3-layer feedforward neural network to map query representations to relevant document representations for RF. Queries and documents are represented with binary vectors using the query and relevant document vocabularies. The learned mapping matrices were then used on test queries to obtain new query representations. Later, Crestni [37] compared this neural RF method with the probabilistic RF models, concluding that this method is less effective. With the progress of hardware that highly improves computation efficiency, more recent studies have explored neural models for RF. Similar to the neural retrieval models introduced in Section 2.1.2, recent studies also explored word embeddings to build RF models or building neural ranking models incorporating feedback information. Rekabsaz et al. [112] built a generalized translation model of BM25 based on cosine similarity between word embeddings and uses the expansion terms generated from Rocchio. Zamani and Croft [161] used the similarity of word embeddings to compute the transition probabilities between words which are then incorporated with the relevance model [79] to solve problems of term mismatch. Similar to the non-neural methods, these RF approaches also try to extract expansion terms. The difference is that they use semantic match at the word level to adjust the probability of choosing expansion terms. Li et al. [81] proposed an end-to-end neural RF framework for document

retrieval, where neural IR models could be used as building blocks and no terms are extracted for expansion. Montazeralghaem et al. [95] explored a reinforcement learning method built on policy networks to estimate expansion term weights from the relevant documents.

2.2.2 Negative Feedback

Negative feedback has been studied mostly together with positive feedback in RF techniques, such as [117, 20, 87, 85, 150, 149]. In [117, 150, 87, 149], a document is ranked based on its dissimilarity to the known non-relevant documents and similarity to relevant documents. Rocchio [117] models feedback in the vector space model, [149] is based on a probabilistic model [151], and the others are based on the language model. Specifically, Ma and Lin [87] extracted positive and negative topic models from both relevant and non-relevant documents. Xu and Akella [149] modeled the positive feedback documents as a mixture of the feedback relevant and background noise models, and the negative feedback documents as a mixture of the feedback relevant, feedback non-relevant, and background noise models. In contrast, Brondwine et al. [20] proposed a distillation model that distills the relevant topic model from the relevant results given the non-relevant ones and ranks documents based on the relevant topic model. However, negative feedback is much less effective than positive feedback in finding more relevant results since the relevant space for a query is much smaller than the non-relevant space [1].

There has been also some research on studying negative feedback alone which mainly focused on document retrieval for difficult queries. Wang et al. [139] proposed to extract a negative topic model from non-relevant documents by assuming that they are generated from the mixture of the topic model of the background corpus and the negative topic model. Wang et al. [140] studied negative feedback methods based on the language model and vector space model (VSM). Later, Karimzadehgan and

Zhai [71] further improved the performance of negative feedback by building a more general negative topic model. These methods typically demote the results similar to the judged non-relevant results and produce limited improvements since the space of non-relevant results is very large. Peltonen et al. [100] introduced a novel search interface, where keyword features of the non-relevant results are provided to users, and they are asked for feedback on the keywords. Then a probabilistic user intent model is estimated to refine re-ranking.

Negative feedback has also been studied for the recommendation task. Zagheli et al. [160] also proposed a language model based method to avoid suggesting results similar to the document users dislike for text recommendation. Zhao Et al. [172] leverage reinforcement learning to learn the optimal strategies by recommending trial-and-error items and collecting feedback from users. Skipped items are considered as negative feedback and combined with positive feedback to facilitate training.

2.2.3 Iterative Feedback

In contrast to most RF systems that ask users to give relevance assessments on a batch of documents, Aalsberg et al. [1] proposed the alternative technique of incremental RF. Users are asked to judge a single result shown in each interaction, then the query model can be modified iteratively through feedback. This approach showed higher retrieval quality compared with standard batch feedback. Later, Lwayama et al. [68] showed that the incremental relevance feedback used by Aalsberg et al. works better for documents with similar topics, while not as well for documents spanning several topics. Allan [12] showed the effectiveness of incremental RF based on Rocchio for information filtering, where documents arrive continuously rather than being static in a collection and queries are long-term rather than one-shot. We plan to investigate how IRF performs on retrieval of answer passages instead of documents using more recent retrieval models.

Some recent TREC tracks [153, 53] have made use of iterative and passage-level feedback, but they focus on document retrieval with different objectives and require a large amount of user feedback. The Total Recall track [153] aims at high recall, where the goal is to promote all of the relevant documents before non-relevant ones. The target of the Dynamic Domain track [53] is to identify documents satisfying all the aspects of the users' information need with passage-level feedback. Both tasks focus on document retrieval and need a large amount of feedback information, which might be impractical in a real search scenario.

In this dissertation, we study neural approaches to iterative positive feedback in answer passage retrieval (Chapter 3), implicit positive feedback in multi-page product search (Chapter 4), leveraging fine-grained feedback when only negative feedback is available in conversational product search (Chapter 5), and asking clarifying questions based on only negative feedback in information-seeking conversations (Chapter 6).

2.3 Conversational Information Retrieval

Conversational IR systems help users find their target information through conversations. User feedback can be potentially obtained during interactions between the system and users naturally. Thus, we based our studies on negative feedback in the scenarios of conversational IR. In the following subsections, we introduce two related research topics in this area: conversational search and question answering, asking clarifying questions.

2.3.1 Conversational Search and Question Answering

The concepts of conversational search were proposed in some earliest work in information retrieval. Oddy [97] first introduced a new method of IR by manmachine interaction in the 1970s. Then, Croft and Thompson [39] designed an intelligent intermediary for information retrieval, called I^3R, which communicates with users during

a search session and reacts based on the goals stated by users and their evaluation of the system output. Belkin et al. [14] built an interactive information retrieval system, MERIT, that used script-based information-seeking dialogues as interaction for effective search.

With the emerging of various intelligent conversational assistants in recent years, task-based conversations based on natural dialogues have drawn much attention. Radlinski and Craswell [108] proposed a theoretical framework with some basic philosophies for conversational information retrieval. Kenter and de Rijke [75] considered building the representation of conversations as the process of machine reading, based on which answers are retrieved. Vtyurina et al. [136] studied how users behave when interacting with a human expert, a commercial intelligent assistant, and a human disguised as an automatic system. Spina et al. [124] studied how to extract audio summaries for spoken document search. Trippas et al. [130] suggested building conversational search systems based on the commonly-used interactions from human communication. Qu et al. [105] collected the conversations from an online forum on Microsoft products and labeled the utterance intents. Yang et al. [156] conducted response ranking based on external knowledge given a conversation history. Aliannejadi et al. [10] created a dataset that consists of information-seeking conversations based on queries with multiple subtopics and proposed to ask clarifying questions to understand user intents better. Wang and Ai [142] propose to control the risk of asking non-relevant questions by deciding whether to ask questions or show results in a conversation turn.

There are also extensive studies on conversational search in recommendation and product search. Mcginty and Smyth [90] leveraged preference and rating based feedback in a conversational recommender system and emphasize product diversity rather than similarity to conduct effective recommendation. Christakopoulou et al. [27] developed a framework to identify which questions to ask in order to quickly learn

user preferences and refine the recommendations during the conversations. Zhang et al. [168] proposed a paradigm for conversational product search, where the system asks users their preferred values of an aspect, shows results when it is confident, and adopts a memory network to ask questions and retrieving results. Sun and Zhang [126] proposed a recommendation system based on a similar paradigm, which also collects users' preferred values for given aspects and uses a reinforcement learning framework to choose actions from asking for the values or making recommendations by optimizing a per-session utility function.

Conversational question answering defines the task of finding an answer span in a given passage based on the question and answers in the conversation history such as CoQA [111] and QuAC [26]. Qu et al. [104] extended the task by introducing a step of retrieving candidate passages for identifying answer span. This is more practical in real scenarios where ground truth passages that contain the answers are often unavailable.

2.3.2 Asking Clarifying Questions

In the TREC 2004 HARD track [13], there has been research on asking searchers clarification questions such as whether some titles seem relevant to improve the accuracy of IR. In recent years, it has drawn much attention to study how to ask clarifying questions in conversational search. Rao et al. [109] collected a clarifying question dataset from the posts in StackOverflow and proposed to select clarification questions based on the expected value of perfect information considering the usefulness of potential answers to a candidate question. Later, Ral et al. [110] extended the work by using the utility [109] in a reinforcement learning framework in product QA to handle cases where contexts such as product information and historical questions and answers are available. In [168, 126], questions about users' preferred values on aspects of a product are asked for conversational product search and recommendation.

Wang et al. [141] observed that a good question is often composed of interrogatives, topic words, and ordinary words and devised typed encoders to consider word types when generating questions. Cho et al. [25] proposed a task of generating common questions from multiple documents for ambiguous user queries. Xu et al. [148] studied whether a question needs clarification and introduced a coarse-to-fine model for clarification question generation in knowledge-based QA systems. Zamani et al. [164] extracted the facets of a query from query logs and generated clarifying questions through template or reinforcement learning with weak supervision.

To study how to ask clarifying questions in open-domain information-seeking conversations, Aliannejadi et al. [10] collected clarifying questions through crowdsourcing in a dataset called Qualc based on the ambiguous or faceted topics in the TREC Web track [29, 30]. They proposed to select the next clarifying question based on BERT representations and query performance prediction. Later, [64] extended the idea of pseudo relevance feedback and leveraged top-retrieved clarifying questions and documents for document retrieval and clarifying question selection on Qulac. Aliannejadi et al. [9] then organized a challenge on clarifying questions for dialogue systems that raises the questions on when to ask clarifying questions during dialogues and how to generate the clarifying questions.

While most existing studies do not differentiate responses that are confirmation or denial, we address how to leverage negative feedback in the response to ask clarifying questions (Chapter 6). Moreover, they evaluate models based on either the initial query or pre-defined conversation history, i.e., the models always select the next question based on static conversation turns instead of its previously selected questions. In contrast, we select the next questions dynamically considering previous questions, which is more consistent to real conversational search scenarios.

2.4 Information Retrieval Applications

IR applications include a wide variety of tasks that seek information such as documents, answers, products, etc. Depending on the information property, there are unstructured and structured data. Documents and answers consist of unstructured text [2], to which term matching and semantic matching are essential during retrieval. In contrast, for structured data like products, various aspects and associated values are the key to differentiate user needs. We briefly review the tasks of text retrieval and product search we base our studies on.

2.4.1 Retrieval of Unstructured Text

Text can be of various lengths such as documents, answers, questions, etc. Text retrieval aims to return texts that are relevant to user queries or questions. Ad-hoc retrieval, answer passage retrieval, open-domain conversational search, clarifying question selection are all examples of text retrieval. Although retrieval models designed for one task can also be used for others, each task has each own focus and usually requires specific care in the model design. We have already introduced conversational search and asking clarifying questions in Section 2.3, so we introduce ad-hoc retrieval and answer passage retrieval in this subsection.

Ad-hoc retrieval is the most well-known text retrieval task and has been studied for many years. It focuses on retrieval documents based on their relevance to queries. The vector space model [120], probabilistic model [88, 113] and language modeling approach for IR [103] introduced in Section 2.1.1 are all originally designed for ad-hoc retrieval. Early retrieval models are mostly based on exact matching of unigrams, bigrams, and trigrams considering term dependency and proximity [103, 52, 91, 15]. Later, by taking semantic matching into account, embedding-based

[2] Although documents could have multi-fields such as title, heading, body, etc. that can be considered as structured, we only discuss the work that treats a document as a whole piece of text and does not consider document structures.

[51, 112, 161, 55, 6] and neural ranking models [54, 147, 98, 50] have drawn considerable attention. They investigated to incorporate the semantic information in the word-based conventional retrieval models or design neural networks to capture the matching patterns between queries and documents.

There has also been extensive research on answer passage retrieval. Initial studies were based on term matching [49, 31, 34] and tried to leverage data from the Web for local corpus [32]. In recent years, retrieval of short text becomes increasingly important in applications such as mobile and voice-based search. Keikha et al. [74, 73] developed a non-factoid answer retrieval dataset, WebAP, using TREC GOV2 collections. They showed that conventional retrieval models that perform well on document retrieval are not effective for this task. Yang et al. [155] proposed to add some semantic and context features in a learning-to-rank framework to retrieve answer sentences for non-factoid questions. Neural ranking models for passage retrieval have also been explored. Wang and Eric [138] trained a bi-directional long-short term memory model (BiLSTM) to retrieve non-factoid answers. Tan et al. [128] employed a convolutional neural network after encoding questions and answers with a BiLSTM. Yang et al. [154] used an attention mechanism to capture matching in the interactions between queries and answers. Cohen and Croft [33] proposed a hybrid approach incorporating embeddings of both words and characters for passage retrieval.

We compare standard top-k RF and iterative RF in both ad-hoc retrieval and answer passage retrieval in Chapter 3 and we further investigate how to improve iterative RF on answer passage retrieval. In the open-domain conversational information seeking task in Chapter 6, we focus on asking clarifying questions based on negative feedback and evaluate the associated ad-hoc retrieval task using the obtained conversations.

2.4.2 Retrieval of Structured Products

Products have the property of entities in contrast to unstructured text. They have common attributes such as brand, price, color, etc., and specific aspects like battery life, screen size, and camera count. Early research mainly leverages facets for product search. Lim et al. [82] proposed a document profile model to annotate the semantic tags of items based on their structured aspects so that the ranking scores can be computed by matching queries with multiple product aspects in the created semantic tags. Vandic et al. [132] studied how to automatically select product facets to minimize the number of steps to find the target product. However, these approaches are for structured queries (e.g., SQL) that are hard and inconvenient for users to write. To support free-from keyword queries, Duan et al. [48, 47] extended language-model-based techniques by assuming that queries are generated from the mixture of one language model of the background corpus and the other one of products conditioned on their specifications. Word mismatch problems still exist in these approaches.

More recent studies consider that product titles, descriptions, and reviews include informative product attributes, and have explored neural models to measure their semantic similarities with queries. Van et al. [131] proposed to map and match n-grams from queries and product descriptions and reviews to the latent semantic space. In contrast to text retrieval where relevance is the most important criterion, in product search, user purchases also depend on their preferences. Aware of the importance of personalization in product search, Ai et al. [7] proposed a hierarchical embedding model based on product reviews and used a convex combination of the query and user vector to predict purchased items. Guo et al. [57] represent long and short-term user preferences with an attention mechanism applied to users' recent purchases and their global vectors. From the analysis of commercial search logs, Ai et al. [4] observed that personalization does not always have a positive effect. They further proposed a zero-attention model (ZAM) that can control the influence of

personalization. However, the maximal effect personalization can have is equal to the query. Bi et al. [18] found this limitation and proposed a transformer model to encode the query and historically purchased items where personalization can have none to full effect. Later, Bi et al. [17] conducted item scoring by dynamically matching user intents and items at the review level instead of explicitly represent them in the semantic space and match them directly.

There are also studies on other aspects such as popularity, other information sources (e.g., images), diversity, and labels for training in product search. Long et al. [83] incorporated popularity with relevance for product ranking. Di et al. [45] and Guo et al. [58] investigated on using images as a complementary signal. Ai et al. [8] proposed an explainable product search model with dynamic relations such as brand, category, also-viewed, also-bought, etc. Efforts have also been made to improve result diversity to satisfy different user intents behind the same query [99, 159]. In terms of training signals, Wu et al. [143] jointly modeled clicks and purchases in a learning-to-rank framework and Karmaker et al. [72] compared the effect of different labels such as click-rate, add-to-cart ratios, and order rates. More recently, Zhang et al. [171] integrated the graph-based feature with neural retrieval models for product search. Xiao et al. [146] studied personalized product search under streaming scenarios. Ahuja et al. [2] learned language-agnostic representations for queries and items that can support search with multiple languages.

In the dissertation, we investigate how to leverage implicit feedback in multi-page product search based on product titles in Chapter 4 and how to incorporate fine-grained feedback on aspect-values using item reviews in conversational product search in Chapter 5.

CHAPTER 3

ITERATIVE RELEVANCE FEEDBACK FOR ANSWER PASSAGE RETRIEVAL

3.1 Introduction

Typical relevance feedback (RF) techniques assume that users provide relevance judgments for the top k (usually 10) documents and then re-rank using a new query model based on those judgments. Even though this is effective, there has been little research recently on this topic because requiring users to provide substantial feedback on a result list is impractical in a typical web search scenario. In new environments such as voice-based search with intelligent assistants, however, feedback about result quality can potentially be obtained during users' interactions with the system. Figure 3.1 shows an example of an assistant interacting with a user collecting her feedback on the provided results. Since there are severe limitations on the length and number of results that can be presented in a single interaction in this environment, the focus should move from browsing result lists to iterative retrieval and from retrieving documents to retrieving answers. In this chapter, we study iterative relevance feedback (IRF) techniques with a focus on answer passage retrieval.

Although they could be applied to any text retrieval scenario, most existing RF algorithms use word-based models originally designed for document retrieval. Answer passages, however, are much shorter than documents, which could potentially present problems for the accurate estimation of word weights in the existing word-based RF methods. Moreover, the limitations on the length and number of results in IRF mean that there is even less relevant text available at every iteration. Given these issues,

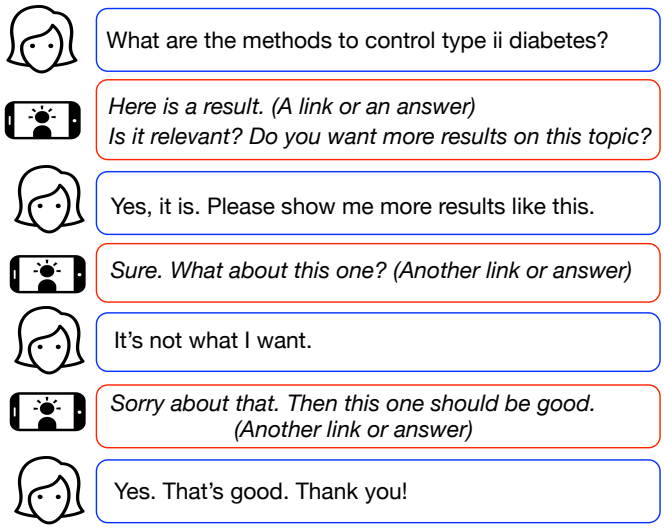


Figure 3.1: An example conversation between a user and an assistant.

introducing complementary information from semantic space may help to estimate a more accurate RF model. Some approaches have incorporated embeddings with RF models [161, 112]. However, these techniques use semantic similarity at the term level and do not consider semantic match at larger granularity. This has led us to incorporate passage-level semantic match in IRF for answer passage retrieval to improve upon word-based IRF and other embedding-based IRF using term-level semantic similarity.

In this chapter, we first convert several representative feedback models to iterative versions. Then we propose an iterative feedback model based on passage-level semantic matching. In the word-based IRF experiments, we show that iterative feedback is comparable with the top-k approach for documents and more effective for answers. In the passage embedding based IRF experiments, we observe that our proposed model grounded on passage-level semantic match produces significant improvements compared to both word-based iterative feedback models and those based on term-level semantic similarity.

3.2 Word-based IRF Models

In IRF, the topic model of users' intent can be refined each iteration after a small number of results are assessed. Therefore, re-ranking is triggered earlier in IRF than in standard top-k RF methods. On the one hand, earlier re-ranking may produce better results with fewer iterations, which essentially reduces the user efforts in search interactions. On the other hand, having only a small amount of feedback information in each iteration may hurt the accuracy of model estimation and cause topic drift in the iterative process, especially for documents that often span several topics.

We convert several representative models to iterative versions and investigate the performance of the IRF models on answer passage retrieval. Since LM and VSM are the two most effective frameworks for RF, [1] we study iterative feedback under these two frameworks. [2] We use RM3 [79] and the Distillation (or Distill) model [20] to represent the LM framework, and Rocchio for VSM. RM3 [79] is a common baseline for pseudo RF that has also been used for RF. Distillation [20] is one of the most recent RF methods, which is an extension of the mixture model [166] by incorporating a query-specific non-relevant topic model. As for the retrieval models for initial ranking, we use Query Likelihood (QL) for LM, and BM25 [116] for VSM respectively. Next, we introduce the iterative versions of RM3, Distillation, and Rocchio.

To keep the query model from diverging to non-relevant topics, we maintain two pools for relevant and non-relevant results, noted as $RP^{(i)}$ and $NRP^{(i)}$, which accumulate all the judgments until the ith iteration. During the ith iteration, judged

[1] The framework we mention in this thesis means the way of doing RF, so the probabilistic model for RF [114, 61] is different from the probabilistic relevance models such as BM25 [116].

[2] We also tried iterative feedback based on the probabilistic model [114], and it showed a similar trend with LM and VSM. However, it generally underperformed the latter two on both document and answer passage retrieval, similar to what Salton et al. [119] reported in their RF experiments, so we do not include it in this dissertation.

relevant results $R^{(i)}$ and non-relevant results $NR^{(i)}$ are added to the corresponding pool. i.e.

$$RP^{(i)} = RP^{(i-1)} \cup R^{(i)}$$

$$NRP^{(i)} = NRP^{(i-1)} \cup NR^{(i)}$$

where $i > 0, RP^{(0)} = \emptyset, NRP^{(0)} = \emptyset$. [3]

We first introduce some notation used in the following formulations. $c(w, x)$ is the count of w in text x; the maximum likelihood estimate (MLE) of term w with respect to a set S is $p_S^{MLE}(w) = \frac{\sum_{x \in S} c(w,x)}{\sum_{x \in S} \sum_{w' \in x} c(w',x)}$; $p_x^{Dir}(w)$ denotes the probability of term w from a Dirichlet smoothed unigram language model induced from x. The relevance score between two language models, $p_{lm_1}(\cdot)$ and $p_{lm_2}(\cdot)$, is calculated with negative KL divergence.

$$\begin{aligned} score_{KL}(p_{lm_1}(\cdot), p_{lm_2}(\cdot)) &= -KL(p_{lm_1}(\cdot)||p_{lm_2}(\cdot)) \\ &\propto \sum_w p_{lm_1}(w) \log p_{lm_2}(w) \end{aligned} \tag{3.1}$$

For RM3 and Distillation, the initial results given the original query $Q^{(0)}$ are ranked with QL according to $score_{KL}(p_{Q^{(0)}}^{MLE}(\cdot), p_x^{Dir}(\cdot))$.

Iterative Relevance Model. Since the true relevant judgments are available, we can estimate the true relevance model in the ith iteration ($i > 1$), directly with binary weights [78, p. 69].

$$p_{rel_{rm3}}^{(i)}(w) = \frac{1}{|RP^{(i-1)}|} \sum_{x \in RP^{(i-1)}} p_x^{MLE}(w) \tag{3.2}$$

[3] We also tried to incrementally represent the query model in the ith iteration with $Q^{(i-1)}$ and $R^{(i-1)}$ and $NR^{(i-1)}$. However, this method performs much worse than using the original query $Q^{(0)}$ and $RP^{(i-1)}$ and $NRP^{(i-1)}$. Given that, we do not include this method in the dissertation.

The updated query language model in the ith iteration is the linear combination of the original query language model and the true relevance model computed from positive feedback:

$$p_{rm3,Q^{(i)}}(w) \stackrel{def}{=} \lambda_{rm3} p^{MLE}_{Q^{(0)}}(w) + (1 - \lambda_{rm3}) p^{(i)}_{rel_{rm3}}(w) \tag{3.3}$$

Then results are ranked with $score_{KL}(p_{rm3,Q^{(i)}}(\cdot), p^{Dir}_x(\cdot))$.

Iterative Distillation Model. The Distillation model assumes that terms in relevant documents are generated from a mixture of a relevance topic model $p_{rel_{distill}}(\cdot)$, a query specific non-relevance topic model $p^{MLE}_{NR}(\cdot)$, and a background corpus language model, $p^{MLE}_{C}(\cdot)$. For the ith iteration ($i > 1$), $p^{(i)}_{rel}(\cdot)$ is estimated with the EM algorithm to maximize the log likelihood of words in $RP^{(i-1)}$.

$$\sum_{x \in RP^{(i-1)}} \sum_{w} c(w, x) \log\big((1 - \lambda_1 - \lambda_2) p^{(i)}_{rel_{distill}}(w) + \lambda_1 p^{MLE}_{NRP^{(i-1)}}(w) + \lambda_2 p^{MLE}_{C}(w)\big) \tag{3.4}$$

Note that if λ_1 is set to 0, Distillation is exactly the same as the mixture model [166]. Similar to RM3, the new query model for the ith iteration is computed as:

$$p_{distill,Q^{(i)}}(w) \stackrel{def}{=} \lambda_{distill} p^{MLE}_{Q^{(0)}}(w) + (1 - \lambda_{distill}) p^{(i)}_{rel_{distill}}(w) \tag{3.5}$$

Then results are ranked with $score_{KL}(p_{distill,Q^{(i)}}(\cdot), p^{Dir}_x(\cdot))$

Iterative Rocchio Model. In VSM, queries and documents are represented with vectors in high-dimensional term space. The weight of each dimension can be calculated in many ways and a similarity measure is used to score documents. In this work, we use the BM25 [116] [4] weight for terms in a document and dot product as

[4]We use the weights calculated from a probabilistic relevance model to represent documents in VSM and use Rocchio as the method for VSM to do RF.

the similarity measure. The BM25 weight is

$$\frac{(k_1+1)\cdot c(w,x)}{k_1(1-b+b\frac{|D|}{avgdl})+c(w,x)}\cdot log\frac{|C|+1}{df(w)} \tag{3.6}$$

where k_1 and b are free parameters. The weight of a query term is set to be $c(w,Q)$. Then in the ith iteration ($i>0$), the query vector can be updated to

$$\vec{Q^{(i)}} = \vec{Q^{(0)}} + \beta\frac{1}{|RP^{(i-1)}|}\sum_{x\in RP^{(i-1)}}\vec{x} + \gamma\frac{1}{|NRP^{(i-1)}|}\sum_{x\in NRP^{(i-1)}}\vec{x} \tag{3.7}$$

If $RP^{(i-1)}$ or $NRP^{(i-1)}$ is empty, the corresponding part is omitted. The relevance score of a document or an answer passage x with respect to a query is computed with the dot product, i.e.

$$score_{VSM}(Q^{(i)},x) = \vec{Q^{(i)}}\cdot\vec{x} \tag{3.8}$$

3.3 Passage Embedding based IRF Models

Word-based RF methods are initially designed for document retrieval and are usually based on query expansion. To conduct effective query expansions, we need sufficient words to cover different aspects of relevant topics and sufficient text to estimate the probabilities or weights of the expansion terms. However, this may be a problem in answer passage retrieval as each result only contains a small amount of text, which might be not enough to build a robust query expansion model. Also, the fact that less relevant results are available in each iteration of IRF makes the problem even worse.

To alleviate the problem of insufficient text in each RF iteration, one possible method is to incorporate semantic information about words and paragraphs. Recent studies have shown that embeddings of words and paragraphs are capable of capturing their semantic meanings [93, 92, 163, 125, 24]. These techniques could potentially help

us build more robust IRF models by supporting semantic matching between words and documents.

Thus we propose to use paragraph embeddings to improve the performance of IRF for answer passage retrieval. In contrast to existing word-based and embedding-based RF methods, this approach does not extract expansion terms to update the query model. Instead, it represents the relevance topic from feedback passages with embeddings. Similar to Rocchio, we assume a relevant passage should be near the centroid of other relevant passages in the embedding space. Also, we only focus on positive feedback as negative feedback has been shown to have little benefit for RF in previous studies [1]. Therefore, our model can be viewed as an embedding version of Rocchio with only positive feedback. In this section, we first describe the methods we use to obtain the semantic representations for answer passages. Then we will introduce the passage embedding based iterative feedback model and discuss other design choices.

3.3.1 Passage Embeddings

Aggregated Word Embeddings. One simple way of representing passages is to use average or weighted average embeddings of words in the paragraph. A popular way of training word embeddings is Word2Vec [92, 93], which projects words to dense vector space and uses a word to predict its context or predicts a word by its context. Representing queries and documents with aggregated word vectors and incorporating similarity of them with a word-based retrieval model has been shown to be useful in cross-lingual IR [137]. For short text like movie reviews, representing them with aggregated word vectors is also common in tasks such as sentiment analysis [80, 24]. So we use aggregated word embeddings trained from Word2Vec both with and without IDF weighting in our experiments.

Paragraph Vectors. The other way of representing passages is using specially designed paragraph vector models as in [80, 24, 125]. The models we use are PV-HDC [125] with or without corruption, shown in Figure 3.2. PV-HDC is an extension of the initially proposed paragraph vector model [80], where a document vector is first used to predict an observed word, and afterward, the observed word is used to predict its context words. The recent work of training paragraph representation through corruption [24] shows advantages in many tasks such as sentiment analysis. It replaces the original part of paragraph representation with a corruption module, where the global context $\tilde{u}$ is generated through an unbiased dropout corruption at each update and the paragraph representation is calculated as the average embeddings of the words in $\tilde{u}$. The final representation is simply the average of the embeddings of all the words in the paragraph. In contrast to [24] which uses the same embeddings for global context (marked in blue) and local context (marked in red), we consider the embeddings for these two types of context can be different. Among our experiments, this way outperforms the original version in most cases. Therefore, we only show experiments under this beneficial setting in Section 3.5. [5]

3.3.2 IRF with Passage-level Semantic Similarity

As an alternative to query expansion-based relevance feedback methods, we propose to represent the whole semantic meaning of a passage and a passage set with vectors in the embedding space, thus the similarity between a relevant passage set and a candidate passage can be calculated without explicitly extracting any expansion terms. Specifically, we represent the relevance topic in the ith iteration as the embedding of the relevant passage pool and fuse the similarity between a candidate

[5]We also investigate other models such as the original PV models, DM, DBOW, [80], and the Parallel Document Context Model (PDC) [125], both with and without corruption. Most models produce improvements, among which HDC is better than other structures in most cases. So we only show the results of PV-HDC/PVC-HDC.

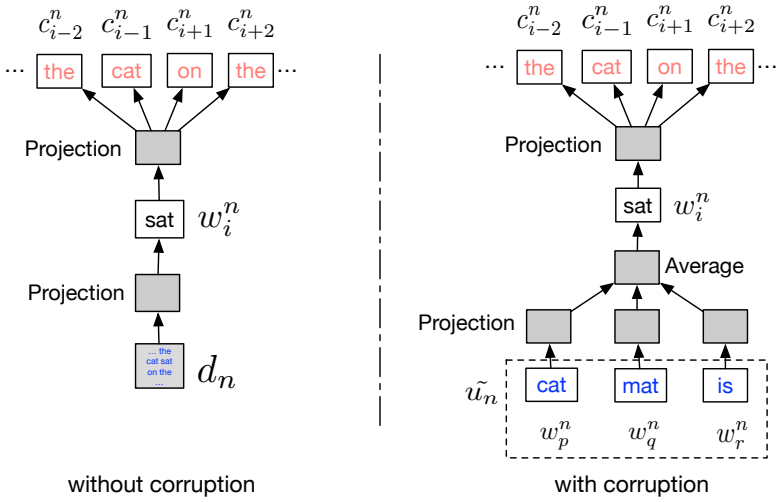


Figure 3.2: HDC models used in our experiments. Red words are local context, and blue words are global context.

passage and the relevance topic with other RF methods. Thus the score function is shown as follows,

$$score(Q^{(i)}, d) = score_{rf}(Q^{(i)}, d) + \lambda_{sf} score_{sem}(RP^{(i)}, d) \tag{3.9}$$

where $Q^{(i)}$ is the expanded query model estimated by iterative version of RF models such as RM3, Distillation and Rocchio; d is the candidate passage; $RP^{(i)}$ denotes the relevant passage pool in the ith iteration; $score_{rf}$ denotes the score calculated from other RF models, such as $score_{KL}$ (Equation (3.1)) or $score_{VSM}$ (Equation (3.8)); λ_{sf} is the coefficient of incorporating the passage embedding based similarity; and $score_{sem}$ is the semantic match score between two pieces of text. Here we choose the commonly used cosine function to measure the similarity. Similar to Rocchio, we assume the topic of a passage set is the centroid of these passages and we consider a relevant passage pool can be represented by weighted average vectors of the passages in it, where the importance of each passage can be estimated according to the quality or the iteration number of it. Thus the similarity between a passage and the pool is

$$score_{sem}(RP^{(i)}, d) = \cos(\sum_{d_r \in RP^{(i)}} w(d_r)\vec{d_r}, \quad \vec{d}) \tag{3.10}$$

where $\vec{d_r}$ is the vector representation in the embedding space, $w(d_r)$ is the weight of each passage. [6] We use uniform weights for each relevant document in the following experiments, and mention some investigation on the effect of assigning weights according to the iteration number in Section 3.5.4.

In addition to this embedding version of Rocchio, we also tried scoring a passage by its average similarity to each relevant passage in the pool. The underlying assumption is that a relevant passage should be similar to each of the other relevant passages. This method can perform significantly better than word-based IRF baselines but is worse than scoring a passage by its similarity with the relevance topic, represented by the centroid of relevant results.

Our method has two advantages over existing RF methods. One is that compared to expansion term based methods that only alleviate word-level mismatch, the semantic similarity of larger granularity is captured in our method. The other is the flexibility of combining this semantic match signal with different types of approaches such as RM3, Distillation, Mixture, Rocchio, and other embedding-based feedback approaches.

3.4 Experiments of Word-based IRF

In this section, we first introduce the experimental setup and then show the results of our word-based IRF in the framework of both LM and VSM on document and answer passage retrieval.

---

[6] We also tried to add the similarity between the original query and a candidate passage in Equation (3.9), but this part does not help in our experiments. One possible reason is that the similarity between a passage with judged relevant passages may already indicate their similarity with the original queries since those relevant passages are judged according to the original queries. The other possible reason is that in the embedding space matching a sentence-level text and a passage-level text is not as accurate as matching two pieces of text at the passage level.

3.4.1 Experimental Setup

Data. In our experiments, we used Robust, Gov2 for document retrieval, and WebAP, PsgRobust for answer passage retrieval. Statistics of the datasets are summarized in Table 3.1. Robust (TREC Robust Track 2004 collection) and Gov2 (TREC Terabyte Track 2004-2006 collection) are two standard TREC collections for ad-hoc retrieval. Robust contains high-quality news articles and Gov2 consists of ".gov" domain web pages. The titles of topics in Robust and Gov2 are used as queries.

WebAP [155] is a web answer passage collection built on Gov2. It selected queries that are likely to have passage-level answers from Gov2 and retrieved the top 50 documents with the Sequential Dependency Model (SDM) [91]. After that, documents identified relevant in the TREC judgments were annotated for answer passages. In our experiments, we used topic descriptions as questions and split the rest of documents randomly into 2-3 sentences as non-relevant passages.

PsgRobust [16] [7] is a new collection we created for answer passage retrieval. It is built on Robust following a similar way as WebAP but without manual annotation. In PsgRobust, we assume that *top-ranked passages in relevant documents can be considered as relevant* and *all passages in non-relevant documents are irrelevant*. We first retrieved the top 100 documents for each title query in Robust with SDM and generated answer passages from them with a sliding window of 2-3 random sentences and no overlap. After that, we retrieved the top 100 passages with SDM again and treated those from relevant documents as the relevant passages. Similar to WebAP, we used the descriptions of Robust topics as questions and have 246 queries with non-zero relevant answer passages in total. The Recall@100 in the initial retrieval process is 0.43, which means that 43% of relevant documents for all queries on average were included in the passage collection on average. Besides, by manually checking some

[7] https://ciir.cs.umass.edu/downloads/PsgRobust/

Table 3.1: Statistics of experimental datasets.

Dataset	#Docs	DocLen	Vocab	#Query	#Qrels
Robust	0.5M	504	0.6M	250	17,412
Gov2	25M	893	35M	150	26,917
WebAP	379k	45	59k	80	3,843
PsgRobust	383k	46	64k	246	6,589

randomly sampled passages marked as relevant, we found most of them are indeed relevant passages for the questions.

We also considered other collections that have passage-level annotation such as the DIP2016Corpus [8] [59] and the dataset from the Dynamic Domain track [153]. However, they either are trivial for relevance feedback tasks (almost all top 10 results retrieved by BM25 are relevant in DIP2016Corpus) or have few queries (only 26 and 27 for the two domains of the Dynamic Domain track). Therefore, we only report the results of WebAP and PsgRobust.

Model Settings. All the methods we implemented are based on the Galago toolkit [38] [9]. Stopwords were removed from all collections using the standard INQUERY stopword list and words were stemmed with Krovetz Stemmer [77]. To compare iterative feedback with typical top-k feedback in a fair manner, we fixed the total number of judged results as 10 and experimented on 1, 2, 5, and 10 iterations, where 10, 5, 2, 1 results were judged during each iteration respectively. Then 10Doc-1Iter is exactly the top-k feedback. We pay more attention to the settings of one or two results per iteration which are more likely to be in a real interactive search scenario considering the limitation of presenting results. True labels of results were used to simulate users' judgments. [10]

[8] `https://github.com/UKPLab/sigir2016-collection-for-focused-retrieval`

[9] http://www.lemurproject.org/galago.php

[10] We did not exclude queries that no relevant results appear in the top 10 retrieved by QL and BM25 since the query sets they perform well at are different and removing those queries will

All the parameters were set using 5-fold cross-validation over all the queries in each collection with grid search, except for retrieval with QL and BM25 on Robust and Gov2, where we use the average value of 5-cross validation from Huston et al's experiments [66]. For WebAP and PsgRobust, we tuned μ of QL in $\{30, 50, 300, 500, 1000, 1500\}$ and k of BM25 from $\{1.2, 1, 4, \cdots, 2\}$, b set to 0.75 as suggested by [94]. We scanned λ_1, λ_2 in Equation (3.4) from $\{0, 0.2, 0.4, \cdots, 1.0\}$ (the sum of λ_1 λ_2 should be less than 1), λ_{rm3} and $\lambda_{distill}$ in Equation (3.3) and (3.5) from $\{0, 0.2, 0.4, \cdots, 0.8\}$, the number of expansion terms m from $\{10, 20, \cdots, 50\}$, and β, γ in equation 3.7 from $\{0, 0.5, 1, \cdots, 3.0\}$.

Evaluation. The evaluation should only focus on the ranking of unseen results. So we use freezing ranking [28, 118], as in [1, 70], to evaluate the performance of iterative feedback. The *freezing ranking* paradigm freezes the ranks of all results presented to the user in the earlier feedback iterations and assigns the first result retrieved in the ith iteration rank $iN + 1$, where N is the number of results shown in each iteration. Note that previously shown results are filtered out in the following retrieval and results retrieved in the last iteration are appended to make a longer rank list. Then we use mean average precision at cutoff 100 (MAP) and $NDCG$@20 to measure the performance of results overall and on the top. As suggested by Smucker et al. [123], statistical significance is calculated with Fisher randomization test with a threshold 0.05.

3.4.2 Results and Discussion

The performance of the initial rank lists retrieved with QL and BM25 are shown in Table 3.2 and the IRF experimental results are shown in Table 3.3. All the values of feedback methods shown in Table 3.3 are significantly better than their retrieval

---

dramatically reduce our query numbers. We keep all the queries to show the results in different frameworks under the same condition. Queries without relevant results in the top 10 will have the same results with RF or not.

baselines, i.e. RM3 and Distillation compared with QL, Rocchio compared with BM25, in terms of both MAP and $NDCG$@20.

IRF on Document Retrieval. As shown in Table 3.3, for document retrieval on Robust, in terms of both metrics, the performance of RM3 and Distillation rises when there are more iterations; the performance of Rocchio drops a little when there are 5 iterations and 10 iterations, but is still similar with that of top-10 feedback. On Gov2, the MAP of RM3 and Distillation with the four iteration settings do not differ much, while the $NDCG$@20 increases as there are more iterations. Rocchio performs the best on 2Doc-5Iter and 1Doc-10Iter in terms of MAP, best on 5Doc-2Iter and 2Doc-5Iter regarding $NDCG$@20. We can see that IRF produces significant improvements in more cases for Robust compared to Gov2 in terms of MAP. The reason may be that Robust is a homogeneous dataset of high-quality news articles and shorter average document length, while Gov2 is a heterogeneous web collection of more various topics and longer average document length. This leads to more possible non-relevant information in the judged relevant documents in Gov2 so topic drift is more likely to happen. Overall, the results of document retrieval show iterative feedback with a relevant document pool does not harm the performance compared with top-k feedback, and improves the performance sometimes, even though fewer documents are used to estimate an accurate relevance topic model.

IRF on Answer Passage Retrieval. Table 3.3 shows, on both answer passage collections, WebAP and PsgRobust, the MAP and $NDCG$@20 of RM3, Distillation and Rocchio increase as the ten results are judged in more iterations. Performance goes up when re-ranking is done earlier even when we have only a small number of passages. [11] The result that IRF performs more effectively in answer passage

[11] On PsgRobust, BM25 and Rocchio underperform QL and Distillation respectively by a large margin. Because the labels of PsgRobust are generated based on the top results retrieved with SDM, this collection favors approaches in the framework of LM more than VSM.

Table 3.2: Performance of QL and BM25.

MAP				
Method	Robust	Gov2	WebAP	PsgRobust
QL	0.215	0.166	0.076	0.248
BM25	0.217	0.167	0.081	0.191
$NDCG$@20				
Method	Robust	Gov2	WebAP	PsgRobust
QL	0.416	0.405	0.143	0.319
BM25	0.418	0.419	0.150	0.292

retrieval than in document retrieval is probably because answer passages are usually focused on a single topic, while documents can span several topics. The non-relevant topics in the relevant documents may cause topic drift compared with answer passages when there are only a small number of feedback results. Since for RM3, Distillation and Rocchio, MAP and $NDCG$@20 show similar trends using iterative feedback on different datasets, we only show MAP in the experiments of embedding-based IRF on answer passage retrieval in Section 3.5.2.

3.5 Experiments of Passage Embedding Based IRF

To show whether our method is effective for IRF on answer passage retrieval, we designed two groups of experiments and have both word-based and embedding-based RF baselines. For the two groups of experiments, one is the same as in Section 3.4, i.e. measuring the performance of each method with a different number of iterations and 10 results judged in total. The other focuses on the ability of each model to identify more relevant passages given only one relevant answer passage. We first describe the experimental setup and then introduce the two groups of experiments in Section 3.5.2 and Section 3.5.3.

Table 3.3: Performance of iterative feedback on document and answer passage collections. All the methods with feedback are significantly better than initial retrieval model (Initial). The initial ranking model is QL for RM3, Distillation, and BM25 for Rocchio. '*' denote significant improvements over the standard top-10 feedback model (10×1) in Fish's randomization test [123] ($p < 0.05$).

Dataset	Method (Doc×Iter)	MAP of freezing rank lists			
		(10×1)	(5×2)	(2×5)	(1×10)
Robust	RM3	0.268	0.274*	0.274*	**0.276***
	Distillation	0.265	0.275*	0.277*	**0.282***
	Rocchio	0.267	**0.273***	0.265	0.266
Gov2	RM3	0.211	0.210	**0.212**	0.211
	Distillation	0.205	0.204	0.204	**0.207**
	Rocchio	0.194	0.196	**0.198***	0.197*
WebAP	RM3	0.100	0.107*	**0.113***	**0.113***
	Distillation	0.099	0.104*	0.109*	**0.111***
	Rocchio	0.106	0.112*	0.118*	**0.119***
PsgRobust	RM3	0.293	0.299*	0.306*	**0.308***
	Distillation	0.292	0.299*	0.311*	**0.313***
	Rocchio	0.268	0.280*	0.285*	**0.286***

Dataset	Method (Doc×Iter)	$NDCG$@20 of freezing rank lists			
		(10×1)	(5×2)	(2×5)	(1×10)
Robust	RM3	0.461	0.474*	**0.478***	**0.478***
	Distillation	0.461	0.474*	0.480*	**0.486***
	Rocchio	0.465	**0.473***	0.462	0.468
Gov2	RM3	0.449	0.459*	0.465*	**0.470***
	Distillation	0.442	0.451*	0.456*	**0.466***
	Rocchio	0.447	**0.459***	0.455*	0.450
WebAP	RM3	0.170	0.180*	0.185*	**0.187***
	Distillation	0.166	0.177*	0.185*	**0.187***
	Rocchio	0.169	0.181*	0.190*	**0.191***
PsgRobust	RM3	0.356	0.363*	0.372*	**0.373***
	Distillation	0.354	0.362*	0.375*	**0.379***
	Rocchio	0.341	0.356*	0.361*	**0.364***

3.5.1 Experimental Setup

In this part, we again use WebAP and PsgRobust for experiments. All comparisons are based on the LM and VSM frameworks. In particular, our methods combined with RM3, Distillation and Rocchio are evaluated to see whether the complementary semantic match information benefits in both frameworks. We also include the Embedding-based Relevance Model (ERM) [161] as a baseline. ERM revises $P(Q|D)$ in the original RM3 as a linear combination of $P(Q|D)$ computed from exact term match and $P(Q|w, D)$, which takes the semantic relationship between words into account. The translation probability between words is computed with the cosine similarity of their embeddings transformed with the sigmoid function. [12] [13]

Embeddings Training. Four paragraph representations are tested in the four groups of experiments, where the base models (BM) can be RM3, ERM, Distillation (or Distill), and Rocchio:

- $BM + W2V / BM + idfW2V$: uniformly or IDF-weighted average word vectors trained with the skip-gram model [92].
- $BM + PV / BM + PVC$: paragraph vectors trained with the HDC structure without corruption or through corruption with separate embeddings for the global context.

[12] The original version of ERM is based on the pseudo relevance feedback version of RM3, which uses the query likelihood score as $P(Q|D)$. Since we do true relevance feedback and use uniform weights for each relevant result in RM3, we revise the $P(Q|D)$ part in ERM to be the same for each relevant result and consistent with the RM3 baseline. In addition, using same weights for each relevant result leads to better performance compared with using pseudo RF version weights.

[13] We also tried another embedding-based baseline, the true RF version of BM25-PRF-GT [112] in the probabilistic relevance framework. It is a generalized translation model of BM25 based on cosine similarity between word embeddings and uses the expansion terms generated from Rocchio. We tried different types of word embeddings, trained from the smaller local corpus or larger global corpus Wikipedia, different dimensions, and different thresholds, but still cannot make it effective on our dataset compared with pure word version RF. So we did not include the experiments here.

Embeddings of words or paragraphs were trained with each local corpus respectively. Words with a frequency less than 5 were removed. No stemming was done across the collections. 10 negative samples were used for each target word. The learning rate and batch size were 0.05 and 256. The dimension of embedding vectors was set to 100. We also tried other hyper-parameters for training embeddings, but the results were similar under different settings. For PVC, corruption rate [24] was set to 0.9. All the neural networks to train embeddings were implemented using TensorFlow [14].

Model Settings. We used the best settings of the baseline models and tuned the parameters of the incorporated semantic signals with 5-fold cross-validation for different paragraph embeddings. All the parameters of ERM are tuned in the same range as [161] suggests. λ_{sf} in equation 3.9 is selected from $\{5, 10, 15, \cdots, 40\}$ for WebAP, and $\{0.5, 1, 1.5, \cdots, 5\}$ for PsgRobust respectively.

3.5.2 Iterative Feedback with Embeddings

First, we conducted IRF experiments with different numbers of iterations and 10 results judged totally, as described in Section 3.4. We use MAP at cutoff 100 of freezing rank lists as the evaluation metric, which is described in section 3.4.1.

Results and Discussion. We show the experimental results of using language model baselines (RM3, ERM, Distillation) in Table 3.4 and include Rocchio as a baseline in Figure 3.3. We can see in general the four representations of paragraphs all can improve performance significantly over the word-based and embedding-based baselines under most iteration settings. ERM performs similarly to RM3 on WebAP, and our method based on RM3 and ERM also perform similarly. On PsgRobust, ERM performs slightly better than RM3 and our method also performs slightly better combined with ERM than RM3. This shows that incorporating passage-level semantic similarity in embedding space produces improvements to both the word-based RF

[14] https://www.tensorflow.org/

Table 3.4: Performance of our proposed method with different paragraph representations compared with word-based and embedding-based RF baselines. '$*$' and '†' denote significant improvements over word-based (RM3, Distillation) and embedding-based (ERM) baselines respectively based on Fisher's randomization test [123] ($p < 0.05$).

Method	MAP on **WebAP**			
(Doc×Iter)	(10×1)	(5×2)	(2×5)	(1×10)
RM3	0.100	0.107	0.113	0.113
ERM	0.101	0.107	0.113	0.116
RM3+W2V	**0.107***†	**0.115***†	0.117	0.116
RM3+idfW2V	0.106*†	0.113*†	0.121*†	0.119*
RM3+PV	0.102*	0.113*†	**0.123***†	**0.123***
RM3+PVC	**0.107***†	0.114*†	0.120*†	0.114
ERM+W2V	**0.107***†	**0.116***†	0.119*†	0.118
ERM+idfW2V	0.106*†	0.114*†	0.121*†	0.118
ERM+PV	0.103*	0.115*†	**0.122***†	**0.121***
ERM+PVC	**0.107***†	0.114*†	0.121*†	0.114
Distillation	0.099	0.104	0.109	0.111
Distill+W2V	**0.106***	**0.114***	**0.120***	0.113
Distill+idfW2V	**0.106***	0.113*	0.116*	0.115
Distill+PV	0.103*	0.110*	0.118*	0.116*
Distill+PVC	0.105*	0.112*	**0.120***	**0.120***

Method	MAP on **PsgRobust**			
(Doc×Iter)	(10×1)	(5×2)	(2×5)	(1×10)
RM3	0.293	0.299	0.306	0.308
ERM	0.294	0.301	0.310	0.310
RM3+W2V	**0.298***†	0.303*†	0.312*†	0.312*
RM3+idfW2V	**0.298***†	0.303*	**0.313***†	0.313*†
RM3+PV	**0.298***†	**0.305***†	**0.313***	**0.314***
RM3+PVC	0.297*†	0.303*	0.308	0.311*
ERM+W2V	**0.299***†	0.304*†	0.313*†	0.312*
ERM+idfW2V	**0.299***†	0.304*†	**0.314***†	**0.314***†
ERM+PV	**0.299***†	**0.307***†	**0.314***†	0.313*
ERM+PVC	0.298*†	0.304*†	0.312*	0.313*†
Distillation	0.292	0.299	0.311	0.313
Distill+W2V	0.297*	0.304*	0.314*	0.319*
Distill+idfW2V	0.297*	**0.306***	0.316*	0.319*
Distill+PV	**0.298***	**0.306***	**0.317***	**0.320***
Distill+PVC	0.297*	0.304*	0.315*	0.317*

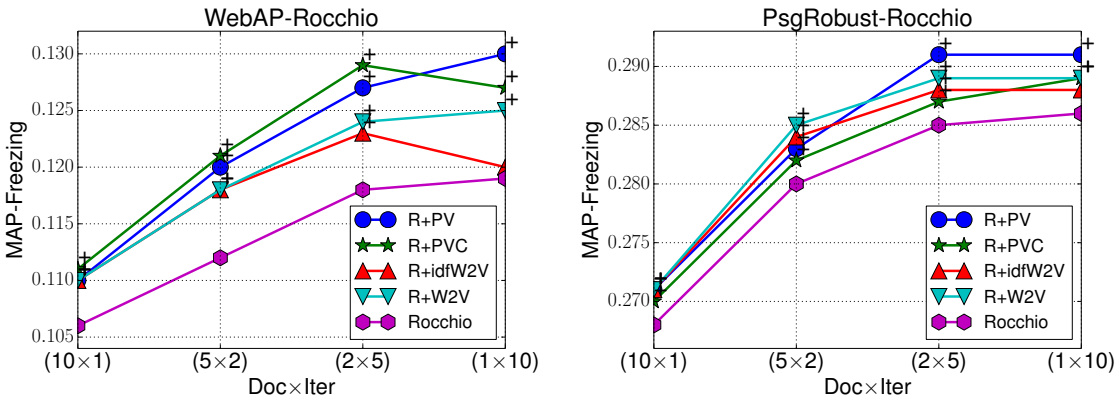


Figure 3.3: Performance of our method with different paragraph representations compared with Rocchio. '+' means significant difference based on Fisher's randomization test [123] ($p < 0.05$)

models and the embedding-based RF model using semantic similarity at the term level. On both WebAP and PsgRobust, PV and PVC improve the performance consistently with different numbers of iterations based on the baseline models. W2V and idfW2V generally perform better than the baselines but not very stably as they show a little disadvantage at 1Doc-10Iter compared with PV and PVC, where sometimes the improvements are not significant. That is probably because paragraph vectors more accurately represent passages as they are trained with the specifically designed structure. There is no one representation better than the others all the time, which implies for different datasets, with different baselines, some representations show their advantages fitting the specific property underlying the setting.

3.5.3 Retrieval Given One Relevant Passage

As we mentioned in Section 3.3, the text in answer passages may not be enough to build word-based RF models, and the small number of results in each iteration make this even more of an issue. The extreme case is when we have only one short passage as positive feedback since non-relevant results do not trigger re-ranking in

the IRF process. Effective re-ranking after *the first positive feedback* will show the user a second relevant answer in fewer iterations. Afterward, it is less difficult to find a third positive answer as new relevant information is added. Therefore, it is particularly important to find another relevant result given the first positive feedback from users. To test our proposed models in this setting, we designed the second type of experiment to be answer retrieval given one relevant passage.

For each query, we randomly assign a relevant passage associated with a query to the model as positive feedback and then retrieval is conducted from the remaining results to see the performance of different models to identify any other correct answer passage. Since we have only a small number of queries, to make the results more reliable without being influenced by random factors, we randomly draw a passage from the relevant passage set for each query ten times and do ten retrievals. Then we evaluate the performance of each model based on the overall rank lists from the ten retrievals. We take QL and BM25 as baseline retrieval models that do not consider feedback. Similar to the first group of experiments, we use RM3, Distillation, Rocchio as word-based RF baselines in the framework of LM and VSM, and ERM as the embedding-based RF baseline. We use P@1 (precision@1), MRR (mean reciprocal rank) to evaluate the ability of a model to retrieve a second relevant passage in the next interaction given only one positive feedback. MAP at cutoff 100 measures the ability of the model to identify all the other relevant answers.

Results and Discussion. In Table 3.5, feedback methods are always better than retrieval models without using feedback, i.e. QL, BM25, by a large margin, as in Section 3.4.2. In general, with the four passage representations, the improvements of MAP over the baselines are always significant; P@1, MRR can also be improved significantly in many cases. This shows that incorporating the passage semantic similarity can improve significantly over both the word-based RF baselines and the embedding-based RF baseline with only term-level semantic match information.

Table 3.5: Performance of different IRF methods on finding other relevant answers given one relevant answer. '$*$' and '†' denote significant improvements over word-based (RM3, Distillation, Rocchio) or embedding-based (ERM) baselines respectively in Fisher's randomization test [123] ($p < 0.05$).

Dataset	**WebAP**			**PsgRobust**		
Model	P@1	MRR	MAP	P@1	MRR	MAP
QL	0.259	0.373	0.071	0.367	0.486	0.231
RM3	0.498	0.602	0.116	0.515	0.634	0.299
ERM	0.516	0.615	0.125	0.513	0.634	0.307
RM3+W2V	0.488	0.598	0.120*	0.524*†	0.643*†	**0.304***
RM3+idfW2V	0.488	0.597	0.120*	0.525*†	0.643*†	**0.304***
RM3+PV	**0.525***	0.625*	0.122*	0.521	0.641*	0.301*
RM3+PVC	0.524*	**0.635***†	**0.123***	**0.526***†	**0.644***†	0.303*
ERM+W2V	0.513	0.622*	0.131*†	0.529*†	0.648*†	0.312*†
ERM+idfW2V	0.525*	0.627*	0.130*†	**0.534***†	0.650*†	0.312*†
ERM+PV	**0.556***†	0.648*†	0.131*†	0.531*†	0.649*†	0.311*†
ERM+PVC	**0.556***†	**0.658***†	**0.134***†	**0.534***†	**0.653***†	**0.313***†
Distillation	0.494	0.597	0.113	0.516	0.635	0.299
Distill+W2V	0.489	0.593	0.117*	**0.528***	**0.645***	**0.304***
Distill+idfW2V	0.489	0.595	0.117*	0.525*	0.643*	**0.304***
Distill+PV	0.519*	0.621*	0.120*	0.514	0.638	0.297
Distill+PVC	**0.534***	**0.638***	**0.123***	0.524*	0.643*	0.303*
BM25	0.298	0.399	0.072	0.350	0.479	0.176
Rocchio	0.516	0.616	0.121	0.522	0.641	0.279
Rocchio+W2V	0.531	0.640*	0.140*	0.526	0.645*	**0.282***
Rocchio+idfW2V	0.536	0.642*	0.139*	0.526	0.644	**0.282***
Rocchio+PV	**0.576***	**0.668***	0.138*	0.518	0.642	0.280*
Rocchio+PVC	0.560*	**0.668***	**0.143***	**0.528***	**0.647***	0.281*

In contrast to the IRF experiments, ERM performs much better than RM3 in this task. One possible reason is that for some queries that need semantic match information, QL cannot retrieve relevant results on the top, which in turn makes it hard for ERM to take effect at earlier iterations. The other reason is that in the IRF experiments, there are more relevant passages for RM3 to extract expansion terms and alleviate the term mismatch problem so that the term-level semantic match from ERM is not very helpful. In this task, the provided information for RM3 is too little to estimate an accurate model and ERM is effective with semantic matching. Since our method considers semantic match at the passage level, it does not overlap with the advantage brought from term-level semantic matching.

On WebAP, our method combined with RM3 performs similarly to ERM when using PV and PVC and worse than ERM using W2V and idfW2V. On PsgRobust, incorporating our method to RM3 performs better than ERM in terms of P@1 and MRR, worse than ERM regarding MAP. This shows incorporating embedding similarity to do RF at passage level or term level alone with extreme little information are comparable to each other. When we combine these two ways of doing RF together, the performance can be further improved, which is shown from the significant improvements upon ERM when we adding the passage similarity signal to ERM on both datasets. This is consistent with our claim that semantic similarity of the passage level is complementary to the term level when combined with word-based RF models since they capture two different granularities of semantic matches.

As in the IRF experiments, the performance of W2V and idfW2V is less stable than that of paragraph vectors. They perform the best on PsgRobust combined with the Distillation model, while in the other cases they sometimes cannot produce significant improvements in terms of P@1 and MRR. Among these representations, PVC performs the most stable, since it outperforms the baselines on both datasets in terms of all three metrics. Overall, it performs better than the other representations

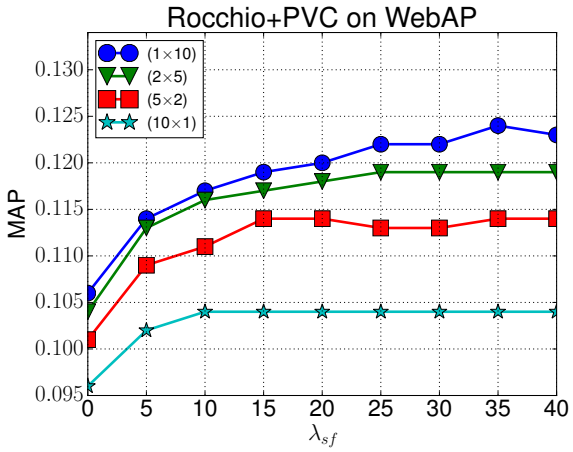
under most settings. This is probably because it provides more accurate passage representation compared with W2V and idfW2V, and less susceptible to overfitting a small dataset compared with PV due to much fewer parameters, i.e. vocabulary size versus corpus size.

3.5.4 Parameter Sensitivity

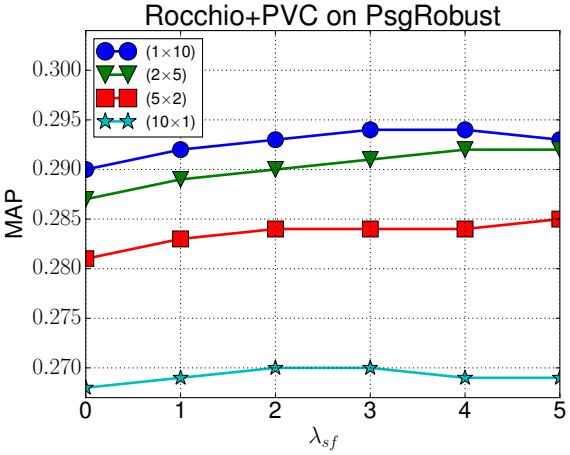
Figure 3.4 shows the sensitivity of performance to λ_{sf} (in Equation (3.9)) on the two groups of experiments. Since different methods and corpus statistics like query length may lead to different score ranges but the cosine similarity between passages is always between -1 and 1, the appropriate ranges of λ_{sf} are different when used with different methods or datasets. In our experiments, the best range of λ_{sf} is from 0 to 5 on WebAP and from 0 to 40 on PsgRobust. Experiments based on Rocchio usually have larger optimal λ_{sf} values compared with language model baselines. For the IRF experiments based on Rocchio, best λ_{sf} is about 35 on WebAP and 4 on PsgRobust, as shown in Figure 3.4. Figure 3.4a and 3.4b again confirm the superiority of IRF over standard top-k feedback. The best λ_{sf} combined with LM-based methods in the IRF experiments is 5 on WebAP and 1 on PsgRobust. We do not include the figures of LM-based methods due to their similar trends to VSM-based methods. For the experiments of retrieval given one relevant passage, the best performance was achieved when λ_{sf} is 5 for LM-based methods, 30 for Rocchio on WebAP, and 1 for LM-based methods, 3 for Rocchio on PsgRobust.

3.6 Summary

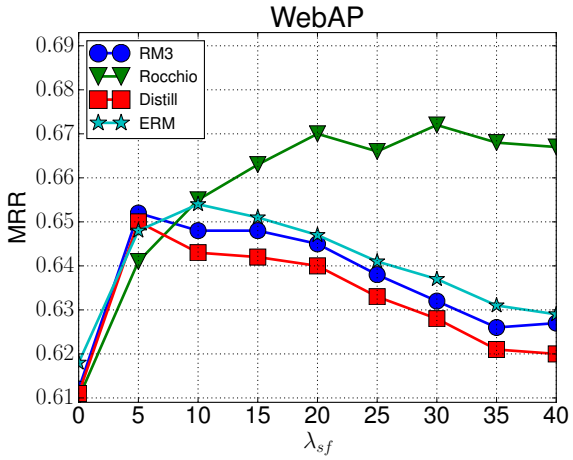
To conclude this chapter, we first investigated the performance of iterative RF on document and answer passage retrieval. Results show that iterative RF is at least as effective as standard feedback on result lists for retrieving documents and is more powerful in finding answers. Then we proposed using passage-level semantic



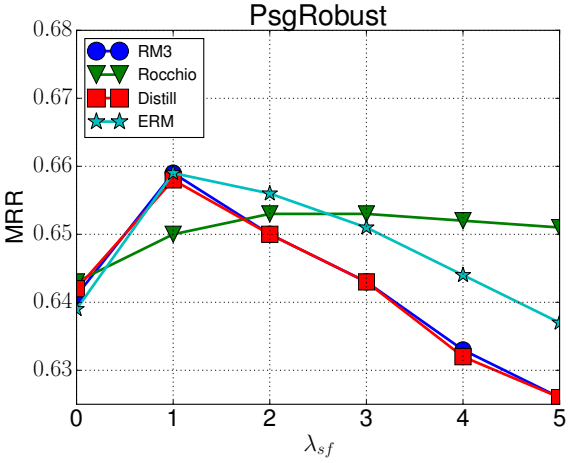
(a) Iterative RF on WebAP



(b) Iterative RF on PsgRobust



(c) Retrieval with one relevant passage



(d) Retrieval with one relevant passage

Figure 3.4: Parameter sensitivity of the coefficient λ_{sf} in Equation (3.9) with PVC on iterative RF and retrieval given one relevant passage.

similarity in iterative RF models, which can be considered as an embedding version of Rocchio. In the IRF experiments, we show that passage-level semantic match produces significant improvements compared to word-based IRF models and other models based on term-level semantic similarity. The retrieval experiment based on one relevant passage shows that in the case where feedback information is scarce, passage-level semantic match is complementary to term-level semantic match and incorporating both of them leads to even better performance.

CHAPTER 4

AN END-TO-END NEURAL RELEVANCE FEEDBACK MODEL FOR PRODUCT SEARCH

4.1 Introduction

Besides iterative answer retrieval, another feasible way to obtain user feedback is to collect user behaviors on search result pages (SERPs) such as clicks and skips. Although noisy, these types of implicit feedback indicate result quality and can be used to refine the ranking of subsequent SERPs. In contrast to web search where users mostly browse results on the first SERPs, in product search, users are more likely to browse results on multiple SERPs before deciding which item to purchase. There has been some research on multi-page search [69, 165], i.e., refining results in the subsequent result pages given users' interactions with results in previous pages. However, these methods are designed for document retrieval, which has different characteristics from product search. Documents consist of text while products are essentially entities that have many aspects such as price, brand, color, and so on. In addition, in contrast to document retrieval, where relevance is a universal evaluation criterion, a product search system is evaluated based on user purchases that depend on both product relevance and customer preferences. Thus, we study relevance feedback models based on user clicks in multi-page product search, where little research has been conducted.

Existing studies have explored eliciting user purchases by providing personalized product search results [7, 18, 17]. They typically leverage user reviews of historically purchased items to capture individual preferences. However, when users have not

logged in or the account is shared by several family members, purchase history will not be available or may be very "noisy". In addition, customers with little or no purchase history do not benefit from personalized product search. Furthermore, preferences extracted from a customer's purchase history are usually long-term and may not always align with her short-term interests. In contrast, feedback models based on user clicks can leverage users' short-term preferences and do not require additional user information or purchase history.

Traditional relevance feedback (RF) methods extract word-based topic models from feedback results as an expansion to the original queries to capture users' preferences. However, they have potential word mismatch problems despite their effectiveness [161, 112]. To tackle this problem, we propose an end-to-end neural feedback model that can incorporate both long-term and short-term context to predict purchased items. In this way, semantic match and the co-occurrence relationship between clicked and purchased items are both captured in the embeddings.

In this chapter, we study feedback models that can use user clicks as positive feedback in context-aware multi-page product search. We first reformulate product search as a dynamic ranking problem, i.e., when users request next SERPs, the remaining unseen results will be re-ranked. We then introduce several context dependency assumptions for the task, which are short-term (user clicks in a query session), long-term (user preferences across all query sessions), and long-short-term. We propose an end-to-end context-aware neural embedding model that can represent each assumption by changing the coefficients to combine long-term and short-term context. We further investigated the effect of several factors in the task: short-term context, long-term context, and neural embeddings. Our experimental results on the datasets collected from Amazon search logs show that incorporating short-term context leads to better performance compared to long-term context and no context, and our neural feedback model is more effective than competitive word-based feedback models.

4.2 Context-aware Product Search

We reformulate product search as a dynamic re-ranking task where short-term context represented by the clicks in the previous SERPs is considered for re-ranking subsequent result pages. Users' global interests can also be incorporated for re-ranking as long-term context. We first introduce our problem formulation and different assumptions of context dependency models. Then we propose a context-aware embedding model for the task and show how to optimize the model.

4.2.1 Problem Formulation

A query session[1] is initiated when a user u issues a query q to the search engine. The search results returned by the search engine are typically grouped into pages with a similar number of items. Let R_t be the set of items on the t-th search result page ranked by an initial ranker and denote by $R_{1:t}$ the union of $R_1, \cdots, R_t$. For practical purposes, we let the re-ranking candidate set D_{t+1} for page $t+1$ be $R_{1:t+k} \setminus V_{1:t}$ where $k \geq 1$ and $V_{1:t}$ is the set of re-ranked items viewed by the user in the first t pages. Given user u, query q, and the set of clicked items in the first t pages $C_{1:t}$ as context, the objective is to rank all, if any, purchased items B_{t+1} in D_{t+1} at the top of the next result page.

4.2.2 Context Dependency Models

There are three types of context dependencies that one can use to model the likelihood of a user purchasing a product in her query session, namely, long-term context, short-term context, and long-short-term context. Figure 4.1 shows the graphical models for these context dependencies, where u denotes the latent variable of a user's long-term interest that stays the same across all the search sessions, and clicks in the

[1]We refer to the series of user behaviors associated with a query as a query session, i.e, a user issues a query, clicks results, turns to other pages, purchases items, and finally ends searching with the query.

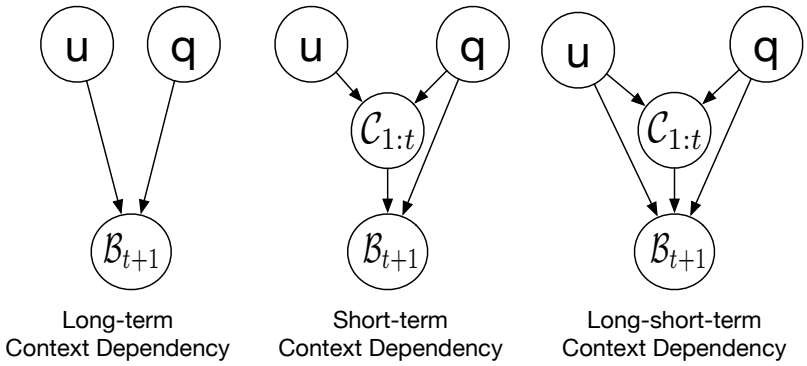


Figure 4.1: Different assumptions to model different factors as context for purchase prediction.

first t result pages, i.e., $C_{1:t}$, represents the user's short-term preference. Purchased items on and after page $t+1$, i.e., $\mathcal{B}_{t+1}$, depends on query q and different types of context under different dependency assumptions.

Long-term Context Dependency. In this assumption, only users' long-term preferences, usually represented by their historical queries and the corresponding purchased items, are used to predict the purchases in their current query sessions. An unshown item i is ranked according to its probability of being purchased given u and q, namely $p(i \in \mathcal{B}_{t+1}|u, q)$. The advantage of such models is that personalization of search results (as proposed in Ai et al. [7]) can be conducted from the very beginning of a query session when there is no feedback information available. However, this model needs user identity and purchase history, which are not always available. In addition, the long-term context may not be informative to predict a user's final purchases since her current search intent may be totally different from any of her previous searches and purchases.

Short-term Context Dependency. The shortcomings of long-term context can be addressed by focusing on just the short-term context, i.e., the user's actions such as clicks performed within the current query session. This dependency model assumes that given the observed clicks in the first t pages, the items purchased in the

subsequent result pages are conditionally independent of the user, shown in Figure 4.1. An unseen item i in the query session is re-ranked based on its purchase probability conditioning on $\mathcal{C}_{1:t}$ and q, i.e., $p(i \in \mathcal{B}_{t+1}|\mathcal{C}_{1:t}, q)$. In this way, users' short-term preferences are captured and their identity and purchase records are not needed. Users with little or no purchase history and who have not logged in can benefit directly under such a ranking scheme.

Long-short-term Context Dependency. The third dependency assumption is that purchases in the subsequent result pages depend on both short-term context, e.g., previous clicks in the current query session, and long-term context, such as historical queries and purchases of the user indicated by u. An unseen item i after page t is scored according to $p(i \in \mathcal{B}_{t+1}|\mathcal{C}_{1:t}, q, u)$. This setting considers more information but it also has the drawback of requiring users' identity and purchase history.

We will introduce how to model the three dependency assumptions in a same framework in Section 4.2.3. In this dissertation, we focus on the case of non-personalized short-term context and include the other two types of context for comparison.

4.2.3 Context-aware Embedding Model

We designed a context-aware framework where models under different dependency assumptions can be trained by varying the corresponding coefficients, shown in Figure 4.2. To incorporate semantic meanings and avoid the word mismatch between queries and items, we embed queries, items, and users into latent semantic space. Our context-aware embedding model is referred to as CEM. We assume users' preferences are reflected by their implicit feedback, i.e. their clicks associated with the query. Similar to relevance feedback approaches [79, 117] that extract a topic model from assessed relevant documents, our model should capture user preferences from their clicked items which are implicit positive signals. Components of CEM will be introduced next.

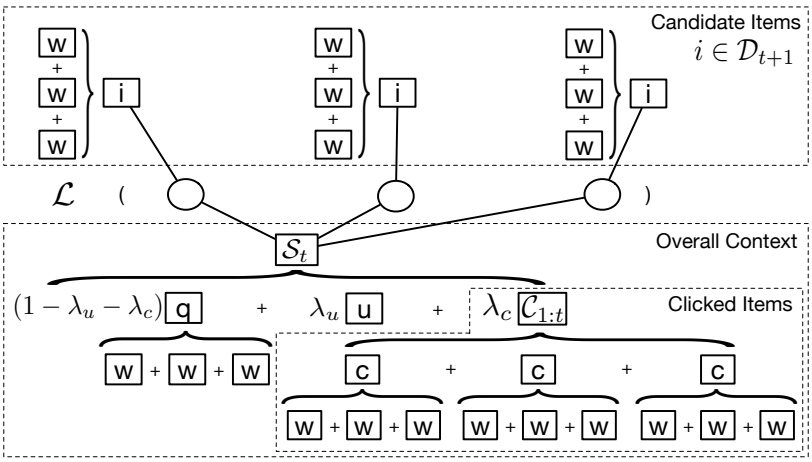


Figure 4.2: The structure of our context-aware embedding model (CEM). w represents words in queries or product titles; $\mathcal{C}_{1:t}$ denotes the click item set in the first t SERPs, which consist of item c; $\mathcal{S}_t$ is the overall context of the first t SEPRs, a combination of query q, user u and clicks $\mathcal{C}_{1:t}$; i is an item in the candidate set $\mathcal{D}_{t+1}$ for re-ranking from page $t+1$.

Item Embeddings. We use product titles to represent products since merchants tend to put the most informative, representative text such as the brand, name, size, color, material, and even target customers in product titles. In this way, items do not have unique embeddings according to their identifiers, and items with the same titles are considered the same. Although this may not be accurate all the time, word representations can be generalized to new items, and we do not need to cope with the cold-start problem. We use the average of title word embeddings of a product as its embedding, i.e.,

$$\mathcal{E}(i) = \frac{\sum_{w \in i} \mathcal{E}(w)}{|i|} \tag{4.1}$$

where i is the item, and $|i|$ is the title length of item i. We also evaluated other more complex product title encoding approaches such as non-linear projection of average word embeddings and recurrent neural network on title word sequence, but they did not show superior performance over the simpler one that we use here.

User Embeddings. A lookup table for user embeddings is created and used for training, where each user has a unique representation. This vector is shared across

search sessions and updated by the gradient learned from previous user transactions. In this way, the long-term interest of the user is captured and we use the user embeddings as long-term context in our models.

Query Embeddings. Similar to item embeddings, we use the simple average embedding of query words as the representation, which also shows the best performance compared to the non-linear projection and recurrent neural network methods we have tried. The embedding of the query is

$$\mathcal{E}(q) = \frac{\sum_{w \in q} \mathcal{E}(w)}{|q|} \tag{4.2}$$

where $|q|$ is the length of query q.

Short-term Context Embeddings. We use the set of clicked items to represent user preference behind the query, which we refer to as $\mathcal{E}(\mathcal{C}_{1:t})$. For sessions associated with a different query q or page number t, the clicked items contained in $\mathcal{C}_{1:t}$ may differ. We assume the sequence of clicked items does not matter when modeling short-term user preference, i.e., the same set of clicked items should imply the same user preference regardless of the order of them being clicked. There are two reasons for this assumption. One is that the user's purchase need is fixed for a query she issued and is not affected by the order of clicks. The other is that the order of user clicks is usually based on the rank of retrieved products from top to bottom as the user examines each result, which is not affected by user preference in the non-personalized search results. So we represent the set as the centroid of each clicked item in the latent semantic space, where the order of clicks does not make a difference. A simple yet effective way is to consider equal weights of all the items in $\mathcal{C}_{1:t}$ so that the centroid is simply averaged item embeddings:

$$\mathcal{E}(\mathcal{C}_{1:t}) = \frac{\sum_{i \in \mathcal{C}_{1:t}} \mathcal{E}(i)}{|\mathcal{C}_{1:t}|} \tag{4.3}$$

where $|\mathcal{C}_{1:t}|$ is the number of clicked items in set $\mathcal{C}_{1:t}$.

We also tried an attention mechanism to weigh each clicked item according to the query and represent the user preference with a weighted combination of clicked items. However, this method is not better than combining clicks with equal weights in our experiments. So we only show simple methods.

Overall Context Embeddings. We use a convex combination of user, query, and click embeddings as the representation of overall context $\mathcal{E}(\mathcal{S}_t)$. i.e.

$$\begin{gathered}\mathcal{E}(\mathcal{S}_t) = (1 - \lambda_u - \lambda_c)\mathcal{E}(q) + \lambda_u \mathcal{E}(u) + \lambda_c \mathcal{E}(\mathcal{C}_{1:t}) \\ 0 \leq \lambda_u \leq 1, 0 \leq \lambda_c \leq 1, \lambda_u + \lambda_c \leq 1\end{gathered} \tag{4.4}$$

This overall context is then treated as the basis for predicting purchased items in $\mathcal{B}_{t+1}$. When $\lambda_c = 0$, $\mathcal{C}_{1:t}$ is ignored in the prediction and $\mathcal{S}_t$ corresponds to the long-term context shown in Figure 4.1. When $\lambda_u = 0$, user u does not have impact on the final purchase given $\mathcal{C}_{1:t}$. This aligns with the short-term context assumption in Figure 4.1. When $\lambda_u > 0, \lambda_c > 0, \lambda_u + \lambda_c \leq 1$, both long-term and short-term context are considered and this matches the type of long-short-term context in Figure 4.1. So by varying the values of λ_u and λ_c, we can use Equation (4.4) to model different types of context dependency and do comparisons.

Attention Allocation Model for Items. With the overall context collected from the first t pages, we further construct an attentive model to re-rank the products in the candidate set $\mathcal{D}_{t+1}$. This re-ranking process can be considered as an attention allocation problem. Given the context that indicates the user's preference and a set of candidate items that have not been shown to the users yet, the item which attracts more user attention will have higher probability to be purchased. The attention weights then act as the basis for re-ranking. Predicting the probability of each candidate item being purchased can be considered as attention allocation for the items. This idea is also similar to the listwise context model proposed by Ai et

al. [3]. They extracted the topic model from top-ranked documents with recurrent neural networks and used it as a local context to re-rank the top documents with their attention weights. The attention weights can be computed as:

$$score(i|q, u, \mathcal{C}_{1:t}) = \frac{\exp(\mathcal{E}(\mathcal{S}_t) \cdot \mathcal{E}(i))}{\sum_{i' \in \mathcal{D}_{t+1}} \exp(\mathcal{E}(\mathcal{S}_t) \cdot \mathcal{E}(i'))} \tag{4.5}$$

where $\mathcal{E}(\mathcal{S}_t)$ is computed according to Equation (4.4). This model can also be interpreted as a generative model for an item in the candidate set $\mathcal{D}_{t+1}$ given the context $\mathcal{S}_t$. In this case, the probability of an item in the candidate set $\mathcal{D}_{t+1}$ being generated from the context $\mathcal{S}_t$ is computed with a softmax function that take the dot product score between the embedding of an item and the context as inputs, i.e,

$$p(i|\mathcal{C}_{1:t}, u, q) = score(i|q, u, \mathcal{C}_{1:t}) \tag{4.6}$$

We need to train the model and learn appropriate embeddings of context and items so that the probability of purchased items in $\mathcal{D}_{t+1}$, namely $\mathcal{B}_{t+1}$, should be larger than the other candidate items, i.e. $\mathcal{D}_{t+1} \setminus \mathcal{B}_{t+1}$. Also, the conditional probability in Equation (4.6) can be used to compute the likelihood of the observed instance of $\mathcal{C}_{1:t}, u, q, \mathcal{B}_{t+1}$.

4.2.4 Model Optimization

The embeddings of queries, users, items are learned by maximizing the likelihood of observing $\mathcal{B}_{t+1}$ given the condition of $\mathcal{C}_{1:t}, u, q$, i.e., after user u issued query q, she clicked the items in the first t SERPs ($\mathcal{C}_{1:t}$), then models are learned by maximizing the likelihood for her to finally purchased items in $\mathcal{B}_{t+1}$ which are shown in and after page $t + 1$. There are many possible values of t even for a same user u if she purchases multiple products on different result pages under query q. These are

considered as different data entries. Then the log likelihood of observing purchases in $\mathcal{B}_{t+1}$ conditioning on $\mathcal{C}_{1:t}, u, q$ in our model can be computed as

$$\begin{aligned}\mathcal{L}(\mathcal{B}_{t+1}|\mathcal{C}_{1:t}, u, q) = \log p(\mathcal{B}_{t+1}|\mathcal{C}_{1:t}, u, q) &\propto \log \prod_{i \in \mathcal{B}_{t+1}} p(i|\mathcal{C}_{1:t}, u, q) \\ &\propto \sum_{i \in \mathcal{B}_{t+1}} \log p(i|\mathcal{C}_{1:t}, u, q)\end{aligned} \tag{4.7}$$

The second step can be inferred if we consider whether an item will be purchased is independent of another item given the context.

According to Equation (4.5), (4.6) and (4.7), we can optimize the conditional log-likelihood directly. A common problem for the softmax calculation is that the denominator usually involves a large number of values and is impractical to compute. However, this is not a problem in our model since we limit the candidate set $\mathcal{D}_{t+1}$ to only some top-ranked items retrieved by the initial ranker so that the computation cost is small.

Similar to previous studies [131, 7], we apply L2 regularization on the embeddings of words and users to avoid overfitting. The final optimization goal can be written as

$$\begin{aligned}\mathcal{L}' =& \sum_{u,q,t} \mathcal{L}(\mathcal{B}_{t+1}|\mathcal{C}_{1:t}, u, q) + \gamma(\sum_{w} \mathcal{E}(w)^2 + \sum_{u} \mathcal{E}(u)^2) \\ =& \sum_{u,q,t} \sum_{i \in \mathcal{B}_{t+1}} \log \frac{\exp(\mathcal{E}(\mathcal{S}_t) \cdot \mathcal{E}(i))}{\sum_{i' \in \mathcal{D}_{t+1}} \exp(\mathcal{E}(\mathcal{S}_t) \cdot \mathcal{E}(i'))} + \gamma\big(\sum_{w} \mathcal{E}(w)^2 + \sum_{u} \mathcal{E}(u)^2\big)\end{aligned} \tag{4.8}$$

where γ is the hyper-parameter to control the strength of L2 regularization. The function accumulates entries of all the possible user u, query q, and the valid page number t. User clicks are on and before page t, and their purchases are after that page. All possible words and users are taken into account in the regularization. When we do not incorporate long-term context, the corresponding parts of u are omitted.

The loss function captures the loss of a list and this list-wise loss is similar to AttentionRank proposed by Ai et al. [3]. Because of the softmax function, optimizing

Table 4.1: Statistics of our collected datasets

	Toys & Games	Garden & Outdoor	Cell Phones & Accessories
Product title length	13.14±6.46	16.39±7.38	22.02±7.34
Vocabulary size	381,620	1,054,980	194,022
Query Session Splits			
Train	91.21%	87.36%	86.57%
Validation	2.61%	3.66%	4.20%
Test	6.18%	8.98%	9.23%

the probabilities of relevant instances in $\mathcal{B}_{t+1}$ simultaneously minimizes the probabilities of the rest non-relevant instances. This loss shows superiority over other list-wise loss functions such as ListMLE [145] and SoftRank [129], which is another reason we adopt this loss.

4.3 Experimental Setup

In this section, we introduce our experimental settings of context-aware product search. We first describe how we construct the datasets for experiments. Then we describe the baseline methods and evaluation methodology for comparing different methods. We also introduce the training settings for our model.

4.3.1 Datasets

We randomly sampled three category-specific datasets, namely, "Toys & Games", "Garden & Outdoor", and "Cell Phones & Accessories", from the logs of a commercial product search engine spanning ten months between years 2017 and 2018. We keep only the query sessions with at least one clicked item on any page before the pages with purchased items. These sessions are difficult for the production model since it could not rank the "right" items on the top so that users purchased items in the second or later result pages. Our datasets include up to a few million query sessions containing several hundred thousand unique queries. When there are multiple purchases in a

query session across different result pages, purchases until page t are only considered as clicks and used together with other clicks to predict purchases on and after page $t+1$. Statistics of our datasets are shown in Table 4.1.

4.3.2 Evaluation Methodology

We divided each dataset into training, validation, and test sets by the date of the query sessions. The sessions that occurred in the first 34 weeks are used for training, the following 2 weeks for validation, and the last 4 weeks for testing. Models were trained with data in the training set; hyper-parameters were tuned according to the model performance on the validation set, and evaluation results on the test set were reported for comparison.

Since the datasets are static, it is impossible to evaluate the models in a truly interactive setting where each subsequent page is re-ranked based on the observed clicks on the current and previous pages. Nonetheless, we can still evaluate the performance of one-shot re-ranking from page $t+1$ given the context collected from the first t pages. In our experiments, we compare different methods for re-ranking from page 2 and page 3 since earlier re-ranking can influence results at higher positions which have a larger impact on the ranking performance. As in relevance feedback experiments [86, 117], our evaluation is also based on residual ranking, where the first t result pages are discarded and re-ranking of the unseen items are evaluated. We use the residual ranking evaluation paradigm because the results before re-ranking are retrieved by the same initial ranker and identical for all the re-ranking methods.

Similar to other ranking tasks, we use mean average precision (MAP) at cut-off 100, mean reciprocal rank (MRR), and normalized discounted cumulative gain ($NDCG$) as ranking metrics. MAP measures the overall performance of a ranker in terms of both precision and recall, which indicates the ability to retrieve more purchased items in the next 100 results and ranking them to higher positions. MRR

is the average inverse rank for the first purchase in the retrieved items. It indicates the expected number of products users need to browse before finding the ones they are satisfied with. $NDCG$ is a common metric for multiple-label document ranking. Although in our context-aware product search, items only have binary labels indicating whether they were purchased given the context, $NDCG$ still shows how good a rank list is with emphasis on results at top positions compared with the ideal rank list. We use $NDCG$@10 in our experiments.

4.3.3 Baselines

We compare our short-term context-aware embedding model (SCEM) with four groups of baseline, retrieval model without using context, long-term, short-term, and long-short-term context-aware models. Specifically, they are:

- **Production Model (PROD).** PROD is essentially a gradient boosted decision tree based model. Comparing with this model indicates the potential gain of our model if deployed online. Note that PROD performs worse on our datasets than on the entire search traffic since we extracted query sessions where the purchased items are in the second or later search result pages.

- **Random (RAND).** By randomly shuffling the results in the candidate set which consists of the top unseen retrieved items by the production model, we get the performance of a random re-ranking strategy. This performance should be the lower bound of any reasonable model.

- **Popularity (POP).** In this method, the products in the candidate set are ranked according to how many times they were purchased in the training set. Popularity is an important factor for product search [83] besides relevance.

- **Query Likelihood Model (QL).** The query likelihood model (QL) [103] is a language model approach for information retrieval. It shows the perfor-

mance of re-ranking without implicit feedback and is only based on the bag-of-words representation. The smoothing parameter μ in QL was tuned from $\{10, 30, 50, 100, 300, 500\}$.

- **Query Embedding based Model (QEM).** This model scores an item by the generative probability of the item given the embedding of a query. When $\lambda_u = 0, \lambda_c = 0$, CEM is exactly QEM.

- **Long-term Context-aware Relevance Model (LCRM3).** Relevance Model Version 3 (RM3) [79] is an effective method for both pseudo and true relevance feedback. It extracts a bag-of-words language model from a set of feedback documents, expands the original query with the most important words from the language model, and retrieves results again with the expanded query. To capture the long-term interest of a user, we use RM3 to extract significant words from titles of the user's historical purchased products and refine the retrieval results for the user in the test set with the expanded query. The weight of the initial query was tuned from $\{0, 0.2, \cdots, 1.0\}$ and the expansion term count was tuned from $\{10, 20, \cdots, 50\}$. The effect of query weight is shown in Section 4.4.2.

- **Long-term Context-aware Embedding Model (LCEM).** When $\lambda_c = 0, 0 < \lambda_u \leq 1$, CEM becomes $LCEM$ by considering long-term context indicated by universal user representations.

- **Short-term Context-aware Relevance Model (SCRM3).** We also use RM3 to extract the user preference behind a query from the clicked items in the previous SERPs as short-term context and refine the next SERP. This method uses the same information as our short-term context-aware embedding model, but it represents user preference with a bag-of-words model and only considers word exact match between a candidate item and the user preference model.

The query weight and expansion term count were tuned in the same range as LCRM3. [2]

- **Long-short-term Context-aware Embedding Model (LSCEM).** When $\lambda_u > 0, \lambda_c > 0, 0 < \lambda_u + \lambda_c \leq 1$, both long-term context represented by u and short-term context indicated by $\mathcal{C}_t$ are taken into account in CEM.

PROD, RAND, POP, QL, and QEM are retrieval models that rank items based on queries and do not rely on context or user information. These models can be used as the initial ranker for any queries. The second type of rankers considers users' long-term interests together with queries, such as LCEM and LCRM3. These methods utilize users' historical purchases but can only be applied to users who appear in the training set. The third type is feedback models which take users' clicks in the query session as short-term context and this category includes SCRM3 and our SCEM. In this approach, user identities are not needed. However, they can only be applied to search sessions where users click on results, and only items from the second result page or later can be refined with the clicks. The fourth category considers both long and short-term contexts, e.g., LSCEM. The second, third, and fourth groups of baseline correspond to the dependency assumptions shown in the first, second and third sub-figure in Figure 4.1 respectively.

4.3.4 Model Training

Query sessions with multiple purchases on different pages are split into sub-sessions, one for each page with a purchase. When there are more than three sub-sessions for a given session, we randomly select three in each training epoch. We do so to avoid skewing the dataset with sessions with many purchases. Likewise, we ran-

[2]We also implemented the embedding-based relevance model (ERM) [161], which is an extension of RM3 by taking semantic similarities between word embeddings into account, as a context-aware baseline. But it does not perform better than RM3 across different settings. So we did not include it.

domly select five clicked items for constructing short-term context if there are more than five clicked items in a query session.

We implemented our models with Tensorflow. The models were trained for 20 epochs with the batch size set to 256. Adam [76] was used as the optimizer and the global norm of parameter gradients was clipped at 5 to avoid unstable gradient updates. After each epoch, the model was evaluated on the validation set and the model with the best performance on the validation set was selected to be evaluated on the test set. The initial learning rate was selected from $\{0.01, 0.005, 0.001, 0.0005, 0.0001\}$. L2 regularization strength γ was tuned from 0.0 to 0.005. λ_q, λ_u in Equation (4.4) were tuned from $\{0, 0.2, \cdots, 0.8, 1.0\}$ ($\lambda_q + \lambda_u \leq 1$) to represent various dependency assumptions mentioned in Section 4.2.2, and the embedding size were scanned from $\{50, 100, \cdots, 300\}$. The effect of λ_q, λ_u and embedding size are shown in Section 4.4.

4.4 Results and Discussion

In this section, we show the performance of the four types of models mentioned in Section 4.3.3. First, we compare the overall retrieval performance of various types of models in Section 4.4.1. Then we further study the effect of queries, long-term context, and embedding size on each model in the following subsections.

4.4.1 Overall Retrieval Performance

Table 4.2 shows the performance of different methods on re-ranking items when users paginate to the second and third SERP for *Toys & Games, Garden & Outdoor* and *Cell Phones & Accessories*. Among all the methods, SCEM and SCRM3 perform better than all the other baselines without using short-term context, including their corresponding retrieval baseline, QEM, and QL respectively, and PROD which con-

[3]Due the confidentiality policy, the absolute value of each metric cannot be revealed.

Table 4.2: Comparison of baselines and our short-term context embedding model (SCEM) on re-ranking when users paginate to the 2nd and 3rd page. The number is the relative improvement of each method compared with the production model (PROD)[3]. '$^-$' indicates significantly worse of each baseline compared with SCEM in student t-test with $p \leq 0.001$. Differences larger than 3% are approximately significant.

Toys & Games						
	Re-ranking from 2nd Page			Re-ranking from 3rd Page		
Model	*MAP*	*MRR*	*NDCG*@10	*MAP*	*MRR*	*NDCG*@10
PROD	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$
RAND	-25.70%$^-$	-26.83%$^-$	-29.23%$^-$	-15.45%$^-$	-17.97%$^-$	-18.96%$^-$
POP	-15.82%$^-$	-15.90%$^-$	-17.87%$^-$	-4.37%$^-$	-5.31%$^-$	-5.18%$^-$
QL	-25.78%$^-$	-27.80%$^-$	-29.73%$^-$	-14.87%$^-$	-18.31%$^-$	-19.20%$^-$
QEM	-2.57%$^-$	-3.10%$^-$	-3.85%$^-$	+12.83%$^-$	+11.07%$^-$	+14.13%$^-$
LCRM3	-24.82%$^-$	-25.92%$^-$	-28.60%$^-$	-13.99%$^-$	-15.82%$^-$	-17.20%$^-$
LCEM	-2.57%$^-$	-3.10%$^-$	-3.85%$^-$	+12.83%$^-$	+11.07%$^-$	+14.13%$^-$
SCRM3	+12.93%$^-$	+9.63%$^-$	+9.53%$^-$	+34.26%$^-$	+29.27%$^-$	+32.86%$^-$
SCEM	**+26.59%**	**+24.56%**	**+26.20%**	**+51.46%**	**+47.57%**	**+54.77%**
LSCEM	**+26.59%**	**+24.56%**	**+26.20%**	**+51.46%**	**+47.57%**	**+54.77%**
Garden & Outdoor						
	Re-ranking from 2nd Page			Re-ranking from 3rd Page		
Model	*MAP*	*MRR*	*NDCG*@10	*MAP*	*MRR*	*NDCG*@10
PROD	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$
RAND	-23.40%$^-$	-24.16%$^-$	-25.73%$^-$	-12.29%$^-$	-13.71%$^-$	-13.97%$^-$
POP	-9.38%$^-$	-9.51%$^-$	-9.55%$^-$	2.09%$^-$	1.43%$^-$	3.49%$^-$
QL	-19.62%$^-$	-20.78%$^-$	-21.63%$^-$	-9.15%$^-$	-10.97%$^-$	-10.37%$^-$
QEM	+0.65%$^-$	-0.34%$^-$	+1.06%$^-$	+15.82%$^-$	+14.42%$^-$	+19.32%$^-$
LCRM3	-19.33%$^-$	-20.45%$^-$	-21.28%$^-$	-9.02%$^-$	-10.73%$^-$	-10.04%$^-$
LCEM	+0.65%$^-$	-0.34%$^-$	+1.06%$^-$	+15.82%$^-$	+14.42%$^-$	+19.32%$^-$
SCRM3	+25.15%$^-$	+23.01%$^-$	+23.15%$^-$	+49.54%$^-$	+46.60%$^-$	+51.20%$^-$
SCEM	**+37.43%**	**+35.16%**	**+37.22%**	**+63.79%**	**+60.43%**	**+67.79%**
LSCEM	**+37.43%**	**+35.16%**	**+37.22%**	**+63.79%**	**+60.43%**	**+67.79%**
Cell Phones & Accessories						
	Re-ranking from 2nd Page			Re-ranking from 3rd Page		
Model	*MAP*	*MRR*	*NDCG*@10	*MAP*	*MRR*	*NDCG*@10
PROD	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$	0.00%$^-$
RAND	-20.15%$^-$	-20.93%$^-$	-22.73%$^-$	-8.75%$^-$	-10.05%$^-$	-9.55%$^-$
POP	-8.54%$^-$	-8.25%$^-$	-11.12%$^-$	-0.78%$^-$	-1.21%$^-$	-1.43%$^-$
QL	-16.14%$^-$	-16.77%$^-$	-18.00%$^-$	-4.05%$^-$	-5.21%$^-$	-3.62%$^-$
QEM	+9.96%$^-$	+9.73%$^-$	+10.58%$^-$	+28.85%$^-$	+27.60%$^-$	+33.92%$^-$
LCRM3	-15.44%$^-$	-16.07%$^-$	-17.38%$^-$	-3.26%$^-$	-4.48%$^-$	-2.85%$^-$
LCEM	+9.96%$^-$	+9.73%$^-$	+10.58%$^-$	+28.85%$^-$	+27.60%$^-$	+33.92%$^-$
SCRM3	+18.65%$^-$	+16.77%$^-$	+17.11%$^-$	+44.52%$^-$	+41.16%$^-$	+46.98%$^-$
SCEM	**+48.99%**	**+47.00%**	**+50.18%**	**+85.51%**	**+81.72%**	**+93.85%**
LSCEM	**+48.99%**	**+47.00%**	**+50.18%**	**+85.51%**	**+81.72%**	**+93.85%**

siders many additional features, showing the effectiveness of incorporating short-term context.

In contrast to the effectiveness of short-term context, long-term context does not help much when combined with queries alone or together with short-term context. LCRM3 outperforms QL on all the datasets by a small margin when users' historical purchases are used to represent their preferences. We varied λ_u from 0 to 1 and found that the best performance was achieved when $\lambda_u = 0$ for LCEM and LSCEM, which means that LCEM and LSCEM always perform worse than QEM and SCEM by incorporating long-term context with $\lambda_u > 0$. Thus, we report the numbers of QEM and SCEM for LCEM and LSCEM respectively to represent their upper-bound performance. Note that since only a small portion of users in the test set appear in the training set, the re-ranking performance of most query sessions in the test set will not be affected. We will elaborate on the effect of long-term context in Section 4.4.3.

We found that neural embedding methods are more effective than word-based baselines. When implicit feedback is not incorporated, QEM performs significantly better than QL, sometimes even better than PROD. When clicks are used as context, with neural embeddings, SCEM is much more effective than SCRM3. This shows that semantic match is more beneficial than exact word match for top retrieved items in product search. In addition, these embeddings also carry the popularity information since items purchased more in the training data will get more gradients during training. Due to our model structure, there are also properties that the embeddings of items purchased under similar queries or context will be more alike compared with non-purchased items, and embeddings of clicked and purchased items are also similar.

The relative improvement of SCEM and SCRM3 compared to the production model on Toys & Games is less than the other two datasets. There are two possible reasons. First, the production model performs better on Toys & Games, compared with Garden & Outdoor, and Cell Phones & Accessories, which can be seen from the

larger advantages compared with random re-ranking. Second, the average clicks in the first two and three SERPs in Toys & Games are less than the other two datasets [4], thus SCEM and SCRM3 can perform better with more implicit feedback information.

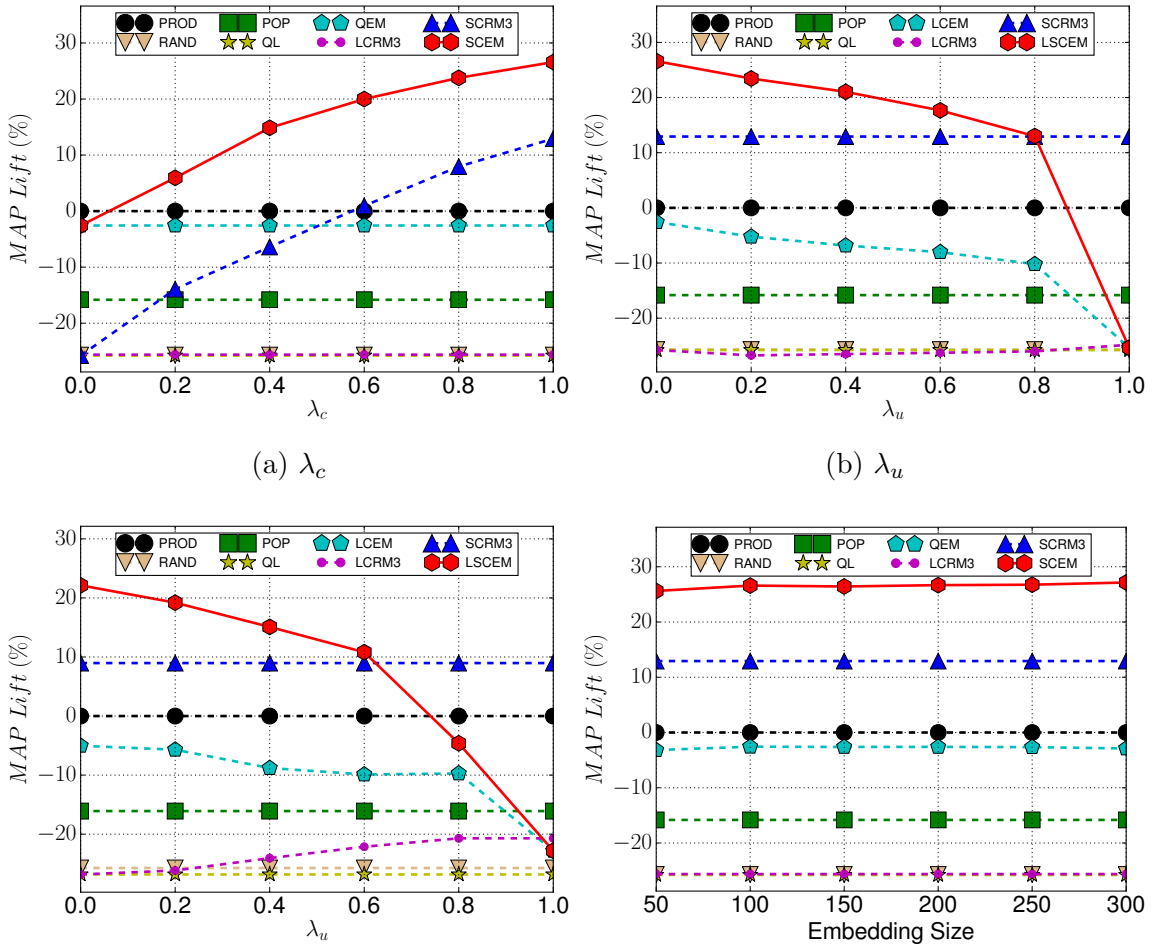
The relative performance of all the other methods against PROD is better when re-ranking from page 2 compared with re-ranking from page 3 in terms of all three metrics. Several reasons are shown as follows. When purchases happen on the third page or later, it usually means users cannot find the "right" products in the first two pages, which further indicates the production model is worse for these query sessions. In addition, the ranking quality of PROD on the third page is worse than on the second page. Another reason that SCRM3 and SCEM improve more upon PROD when re-ranking from page 3 is that more context becomes available with clicks collected from the second page and makes the user preference model more robust.

QL performs similarly to RAND on Toys & Games and a little better than RAND on Garden & Outdoor, and Cell Phones & Accessories, which indicates that relevance captured by exact word matching is not the key concern in the rank lists of the production model. In addition, most candidate products are consistent with the query intent but the final purchase depends on users' preference. Popularity, as an important factor that consumers will consider, can perform better than QL. However, it is still worse than the production model most of the time.

4.4.2 Effect of Short-term Context

We investigate the influence of short-term context by varying the value of λ_c with λ_u set to 0. The performance of SCRM3 and SCEM varies as the interpolation coefficient of short-term context changes since only these two methods utilize the

[4]The specific number of average clicks in the datasets cannot be revealed due to the confidentiality policy.



(a) λ_c

(b) λ_u

(c) λ_u on seen users

(d) Embedding size

Figure 4.3: The effect of λ_c, λ_u, embedding size on the performance of each model in the collection of *Toys & Games* when re-ranking from the second SERP for the scenarios where users paginate to page 2.

clicks. Since re-ranking from the second or third pages on *Toys & Games, Garden* and *Mobile* all show similar trends, we only report the performance of each method in the setting of re-ranking from second pages on *Toys & Games*, which is shown in Figure 4.3a. Figure 4.3a shows that as the weight of clicks is set larger, the performance of SCRM3 and SCEM goes up consistently. When λ_c is set to 0, SCRM3 and SCEM degenerate to QL and QEM respectively which do not incorporate short-term context. From another perspective, SCRM3 and SCEM degrade in performance as we increase the weight on queries. For exact word matching based methods, more click signals lead to more improvements for SCRM3, which is also consistent with the fact that QL performs similarly to RAND by only considering queries. For embedding-based methods which capture semantic match and popularity, QEM with queries alone performs similarly to PROD but much better when more context information is incorporated in SCEM. This indicates that users' clicks already cover the query intent, and also contain additional users' preference information.

4.4.3 Effect of Long-term Context

Next, we study the effect of long-term context indicated by users' global representations $\mathcal{E}(u)$ both with and without incorporating short-term context. QEM and LCRM3 only use queries and user historical transactions for ranking; LSCEM uses long and short-term context ($\lambda_u + \lambda_c$ is fixed as 1 since we found that query embeddings do not contribute to the re-ranking performance when short-term context is incorporated). *Toys & Games* is used again to show the sensitivity of each model in terms of λ_u under the setting of re-ranking from the second page. Since there are users in the test set who never appear in the training set, λ_u does not take effect due to the null representations for these unknown users. In Toys & Games, only about 13% of all the query sessions in the test set are from users who also appear in the training set. The performance change on the entire test set will be smaller due

to the low proportion the models can affect in the test set, so we also include the performance of each model on the subset of data entries associated with users seen in the training set. Figure 4.3b and 4.3c show how each method performs on the whole test set and the subset respectively with different λ_u.

Figure 4.3b and 4.3c show that for LSCEM, as λ_u becomes larger, performance goes down. This indicates that when short-term contexts are used, users' embeddings act like noise and drag down the re-ranking performance. λ_u has different impacts on the models not using clicks. For LCRM3, when we zoom in to only focus on users that appear in the training set, the performance changes and the superiority over QL are more noticeable. The best value of MAP is achieved when $\lambda_u = 0.8$, which means long-term context benefit word-based models with additional information, which can help solve the word mismatch problem. In contrast, for LCEM, with non-zero λ_u, it performs worse than only considering queries. Embedding models already capture semantic similarities between words. In addition, as we mentioned in Section 4.4.1, they also carry information about popularity since the products purchased more often under the query will get more credits during training. Another possible reason is that the number of customers with sessions of similar intent is low so that the user embedding is misguiding the query sessions. Thus, users' long-term interests do not bring additional information to further improve LCEM on the collections.

This finding is different from the observation in HEM proposed by Ai et al. [7], which incorporates user embeddings as users' long-term preferences and achieves superior performance compared to not using user embeddings. We hypothesize that this inconsistent finding is due to the differences in datasets. HEM was experimented on a dataset that is heavily biased to users with multiple purchases and under a rather simplistic assumption of query generation, where the terms from the category hierarchy of a product are concatenated as the query string. Their datasets contain only hundreds of unique queries and tens of thousands of items that are all purchased by

multiple users. In contrast, we experimented on the real queries and corresponding user behavior data extracted from real search logs. The number of unique queries and items in our experiments is hundreds of times larger than in their dataset. There is also little overlap of users in the training and test set in our datasets, while in their experiments, all the users in the test set are shown in the training set.

4.4.4 Effect of Embedding Size

Figure 4.3d shows the sensitivity of each model in terms of embedding size on *Toys & Games*, which presents similar trends to the other two datasets. Generally, SCEM and QEM are not sensitive to the embedding size as long as it is in a reasonable range. To keep the model effective and simple, we use 100 as the embedding size and report experimental results under this setting in Table 4.2 and the other figures.

4.5 Summary

To sum up, we study how to use user clicks as implicit feedback to refine ranking in the subsequent result pages in multi-page product search. We observe that short-term context, i.e., user clicks in the query session, is much more beneficial than long-term context in multi-page product search. Also, experimental results show that our end-to-end embedding-based feedback model is effective to incorporate the implicit feedback, indicating that averaging item embeddings is an effective way to represent user preferences (or feedback topics). This idea is simple but very effective, which indicates that feedback models do not need to be in the form of query expansion. For a neural retrieval model, the query representation could be combined with the representation of results with positive feedback as a new query model, where no explicit terms need to be extracted to expand the original query.

CHAPTER 5

CONVERSATIONAL PRODUCT SEARCH BASED ON NEGATIVE FEEDBACK

5.1 Introduction

In Chapter 3, we focus on retrieving more relevant answer passages given some known relevant passages judged by users in an iterative way. This is beneficial for the scenario where multiple relevant answers are needed to satisfy the users' needs. In a more general scenario, one relevant answer will often satisfy the users' need and thus iterative relevance feedback would only bring marginal benefits by showing extra relevant results. In this case, only negative feedback is available to help retrieve relevant results in the next iteration. In this chapter, we will study how to retrieve the first relevant result based on non-relevant results.

In contrast to text retrieval, the search space for product retrieval is much smaller since there are a limited number of products while much more unstructured text can be formed as candidates for text retrieval. In addition, products are more structured, which have aspect-values such as "price-high", "brand-apple" and so on. Moreover, users usually just buy one item given a single purchase need. These properties make it appropriate and potentially easier to study negative feedback on product search. So we focus on the retrieval of products as the first step to study negative feedback in information retrieval.

Compared with positive feedback, negative feedback is more challenging to refine re-ranking since relevant results usually have similar characteristics while the reason for a result to be non-relevant could be varied. Previous work on negative feedback

[71, 139, 140] mainly focuses on document retrieval for difficult queries. They extract negative topic models from the non-relevant documents and demote the results with high similarities to the negative topic models during re-ranking. However, result-level negative feedback is not very informative especially when there are only a few non-relevant results available. In product search, when a user does not like an item, it is easy to collect their detailed feedback on certain properties (aspect-value pairs) of the item. By breaking down the item-level negative feedback to fine-grained feedback on aspect-value pairs, more information is available to help identify user preferences and thus can be promising to provide better-tailored results.

In this chapter, we study how to leverage negative feedback in the scenario of conversational product search. We first propose a conversation paradigm for product search driven by items with negative feedback, based on which the system further collects fine-grained feedback for ranking refinement in the next iteration. We then propose an aspect-value likelihood model to incorporate both positive and negative feedback on fine-grained aspect-value pairs of the non-relevant items. Experimental results show that our model is significantly better than state-of-the-art product search baselines without using feedback and baselines using item-level negative feedback.

5.2 Conversation Paradigm and Problem Formulation

The paradigm we propose for conversational product search motivated by negative feedback is shown in Figure 5.1. After the user's initial request, several items are shown to the user. If she is not satisfied with the items, her detailed preferences on aspect-value pairs (such as "battery-removable") of the items are gathered. Then based on the fine-grained feedback on the non-relevant results, the remaining items are re-ranked in the next iteration. This process proceeds until the user finally finds the "right" product. The whole process is formalized as follows.

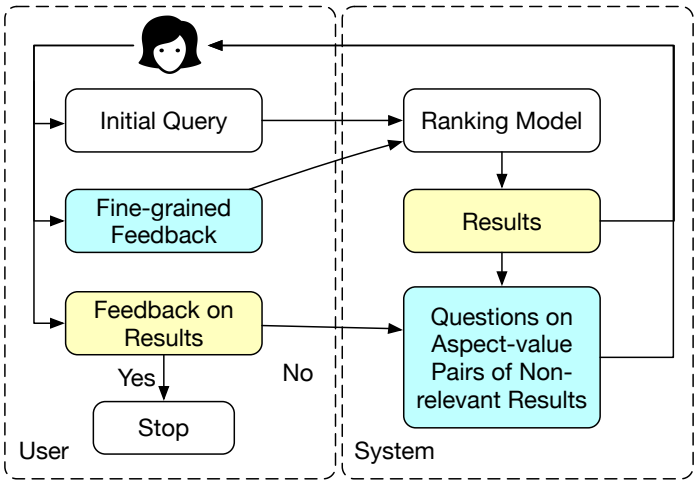


Figure 5.1: A workflow of conversational search system based on negative feedback.

A conversation is initiated with a query Q_0 issued by a user u. In the k-th iteration, a batch of results D_k are retrieved and shown to the user. When D_k does not satisfy the user need, from all the shown non-relevant results, $D_1 \cup D_2 \cdots \cup D_k$, denoted as $D_{1:k}$, the system extracts a set of aspect-value pairs, namely, $AV(D_{1:k})$. Then the system selects m aspect-value pairs $\{(a_{k,j}, v_{k,j})|1 \leq j \leq m\}$ from $AV(D_{1:k})$ and asks m corresponding questions $\{\mathcal{Q}(a_{k,j}, v_{k,j})|1 \leq j \leq m\}$ to the user about whether he/she likes the aspect-value pairs of the non-relevant results. After collecting the user's feedback to $\mathcal{Q}(a_{k,j}, v_{k,j})$, denoted as $I(a_{k,j}, v_{k,j})$, in the $k+1$-th iteration, the goal of the system is to show a list of results D_{k+1}, which ranks the finally purchased item i on the top. The sequence of actions in the conversation can be represented with

$$u \rightarrow Q_0; D_1, \mathcal{Q}_{1,1}, I_{1,1}, \cdots, \mathcal{Q}_{1,m}, I_{1,m}; \cdots;$$

$$D_k, \mathcal{Q}_{k,1}, I_{k,1}, \cdots, \mathcal{Q}_{k,m}, I_{k,m} \rightarrow i$$

where $\mathcal{Q}_{k,j}$ and $I_{k,j}$ denote $\mathcal{Q}(a_{k,j}, v_{k,j})$ and $I(a_{k,j}, v_{k,j})$ respectively. $\mathcal{Q}_{k,j}$ is a yes-no question and $I_{k,j}$ can be 1 or -1 to indicate that the answer is yes or no to the question. In addition, reviews of u and i are available to facilitate the ranking, denoted as R_u and R_i respectively.

In this chapter, we focus on the scenario where only one result is retrieved during each iteration, namely $|D_k| = 1$. However, the method we propose can cope with general cases with more than one result retrieved in each iteration.

5.3 Aspect-value Likelihood Embedding Model

There are two major modules in our system to conduct product search through conversations with users: selecting aspect-value pairs to ask for feedback and ranking based on the fine-grained feedback. For the aspect-value pair selection, we adopt heuristic strategies, i.e., selecting several random pairs, or pairs mentioned most in the reviews of the non-relevant items, and leave the investigation of other potentially better methods as future work. Then we focus on the ranking model that leverages feedback on aspect-value pairs. We propose an aspect-value likelihood embedding model (AVLEM) which can rank items both with and without feedback. The overall structure of AVLEM is shown in Figure 5.2. We introduce each component of AVLEM in the following subsections.

5.3.1 Item Generation Model

We construct an item generation model to capture the purchase relationship between items and their associated users and queries. Similar to [7], an item i is generated from a user u and her initial request query Q_0. The probability can be computed with the softmax function on their embeddings:

$$P(i|u, Q_0) = \frac{\exp\Big(\mathbf{i} \cdot \big(\lambda \mathbf{Q_0} + (1-\lambda)\mathbf{u}\big)\Big)}{\sum_{i' \in S_i} \exp\Big(\mathbf{i'} \cdot \big(\lambda \mathbf{Q_0} + (1-\lambda)\mathbf{u}\big)\Big)} \tag{5.1}$$

where S_i is the set of all the items in the collection, λ is the weight of the query in the linear combination. The representations of Q_0, u and i will be introduced next.

5.3.2 Query Representation

In order to generalize the representations to unseen queries, we use the embedding of query words as input and adopt a non-linear projection of the average word embeddings as the representation of a query:

$$\mathbf{Q_0} = f(\{w_q | w_q \in Q_0\}) = \tanh(W \cdot \frac{\sum_{w_q \in Q_0} \mathbf{w_q}}{|Q_0|} + b) \tag{5.2}$$

where $W \in \mathbb{R}^{d \times d}$ and $b \in \mathbb{R}^d$ when the size of embeddings is d, $|Q_0|$ is the length of query Q_0. This method has been shown to be more effective in [7] compared with using average embeddings of words and a recurrent neural network to encode the word embedding sequence in the query for product search. This finding is different from what we observe in Chapter 4, probably due to the different characteristics of the simulated queries in our experimental datasets compared to real search queries in the previous chapter. For this task, we constructed queries from concatenation of item categories, e.g., "cell phone accessory international charger", "cell phone accessory case sleeve", "store kindle ebook cookbook food wine bake dessert", which are about 2 to 3 words longer than real queries on average. The averaging approach works better than the projection on short fluent real queries but not as effective on longer concatenated simulated queries.

5.3.3 User/Item Language Model

To alleviate the potential vocabulary mismatch between queries and items, we also adopt the user/item language model in [7] to learn the representation of users and items by constructing language models from their associated reviews. Words in the reviews are assumed to be generated from a multinomial distribution of a user or an item. Take user u for example, given its embedding $\mathbf{u}$ ($\mathbf{u} \in \mathbb{R}^d$) and the embedding of

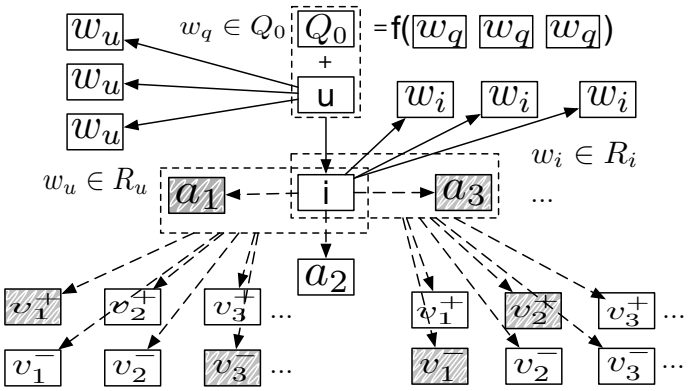


Figure 5.2: The architecture of our aspect-value likelihood embedding model (AVLEM). The solid and dotted arrows represent the generation from a multinomial and a multivariate Bernoulli distribution respectively. The shaded and blank background represents the occurrence and nonoccurrence of the target. v^+ and v^- denote positive and negative values.

a word w, $\mathbf{w}(\mathbf{w} \in \mathbb{R}^d)$, the probability of w being generated from the language model of u is defined with a softmax function on $\mathbf{w}$ and $\mathbf{u}$:

$$P(w|u) = \frac{\exp(\mathbf{w} \cdot \mathbf{u})}{\sum_{w' \in S_w} \exp(\mathbf{w}' \cdot \mathbf{u})} \tag{5.3}$$

where S_w is the vocabulary of words in the reviews from the corpus. Similarly, the language model for item i is represented with $p(w|i)$, which is the softmax over $\mathbf{w}$ and $\mathbf{i}$. Words are assumed to be generated from the language models of user and items independently.

5.3.4 Aspect-Value Generation Model

In addition to the common aspects shared by all the items such as "brand", "color", and "price", there are specific aspects that pertain to certain products such as "battery life", "screen size", and "collar shape". Thus, the probabilities of various aspects associated with an item are different. To capture the probability of an aspect-value associated with an item, we propose an aspect-value generation model, which can be further decomposed to aspect generation given an item and value generation

given an aspect and an item. Both positive and negative feedback on aspect-value pairs are incorporated into the model. We first show the assumptions of multivariate Bernoulli distributions for generating aspects and values. Then we show how we construct aspect-value embeddings and learn them in the aspect-value generation model.

Multivariate Bernoulli (MB) Assumption for Aspects. We propose a multivariate Bernoulli model for aspect generation. Given a purchased item, aspects of the item are assumed to be generated from n_a independent Bernoulli trials of n_a aspects, where n_a is the total number of available aspects and each aspect may have a different probability of appearing in the item's associated aspects. The associated aspects can be any reasonable aspect of the item, e.g., aspects collected from the item's meta-data or reviews. Another possible assumption is the multinomial distribution, which is commonly used to model the documents being generated from words in the vocabulary, such as in the query likelihood model [103]. However, this assumption is not appropriate for aspect generation because aspects are not exclusive and the probabilities of all the aspects generated from one item are not necessarily summed to 1. For example, for an item, "style", "appearance", and "material" are not mutually exclusive. The higher probability of "style" should not lead to the lower probability of "appearance" or "material". So the MB model is more reasonable by considering these aspects generated independently during their own Bernoulli trial.

Multivariate Bernoulli Assumption for Values. Similar to aspect generation, the values of an item's aspect are also assumed to be generated from a MB distribution instead of a multinomial distribution. The property that probabilities of all the values given an item's aspect are summed to 1 is not suitable, especially for values with negative feedback. For example, the aspect, "battery life", of an item can be "short" or "terrible", and a user shows that she does not want the battery life to

be short. Minimizing the probability of her ideal item's "battery life" to be "short" in a multinomial model may lead to a higher probability of "terrible".

Instead of modeling the generation of values with one MB distribution, we propose two independent MB models for the generation of values in positive and negative feedback respectively. Positive values are assumed to be generated from n_v independent trials of n_v values and each value has its own probability of appearing in positive values. Negative values are assumed to be generated from a similar process based on its own MB model. This approach is more reasonable because values without positive feedback are not necessarily disliked by a user and values on which the user has not provided negative feedback are not necessarily liked. A value could be valid for the item's aspect but does not receive positive or negative feedback since the system has not asked for feedback on this value, or the user has vague opinions towards the value. Our experiments also show better performance of having a separate MB model for negative values compared with using one MB model for both positive and negative values in Section 5.5.2.

Aspect and Value Embeddings. Words contained in the aspects and values are also in the vocabulary of words in reviews. Since these words represent the characteristics of items, different from words in the reviews that are generated from the item language model, we keep separate embedding lookup tables for the words in the vocabulary of aspects and values to differentiate the properties of the same words in the aspect-value pairs or item reviews.

Aspects of an item can be of multiple words, such as"battery life" and "touch screen", so we also adopt Equation (5.2) and compute the embedding of an aspect a as $\mathbf{a} = f(\{w_a | w_a \in a\})$. Positive values and negative values have two separate groups of embeddings so that values have different representations in the MB models for positive and negative values. Since values usually consist of one word, such as "long", "big", "clear", and "responsive", the embedding of a value v is just its word

embedding, i.e., $\mathbf{v}^+$ for v in the positive values, and $\mathbf{v}^-$ for v in the negative values. Note that these two embeddings are different from the representation of v as a word in the reviews, and values with more than one word were removed from the corpus.

Aspect-Value Probability Estimation. Next, we show how to estimate the probabilities in the multivariate Bernoulli models of aspects and values. Given the embedding representation of items, aspects and values, the probability of aspect a occurring in the reviews given an item i is

$$P(a \in A(i)|i) = \delta(\mathbf{a} \cdot \mathbf{i}) \tag{5.4}$$

where $A(i)$ is the set of aspects of i, and δ is the sigmoid function $\delta(x) = \frac{1}{1+e^{-x}}$; the probability that value v occurs in the positive value set of item i's aspect a, i.e., $\{v|I(a,v) = 1\}$, denoted by $V^+(i,a)$, is

$$P(v \in V^+(i,a)|i,a) = \delta(\mathbf{v}^+ \cdot (\mathbf{i} + \mathbf{a})) \tag{5.5}$$

where $\mathbf{v}^+$ is the embedding of v as a positive value. Then the probability that an aspect-value pair (a,v) appears in users' positive feedback given an item i can be computed as:

$$\begin{aligned} P(I(a,v) = 1|i) &= P(v \in V^+(i,a)|i,a)P(a \in A(i)|i) \\ &= \delta(\mathbf{v}^+ \cdot (\mathbf{i} + \mathbf{a})) \cdot \delta(\mathbf{a} \cdot \mathbf{i}) \end{aligned} \tag{5.6}$$

Similarly, the probability that (a,v) occurs in the negative feedback in a conversation that leads to purchasing item i, i.e., $P(I(a,v) = -1|i)$ can be calculated according to:

$$\begin{aligned} P(I(a,v) = -1|i) &= P(v \in V^-(i,a)|i,a)P(a \in A(i)|i) \\ &= \delta(\mathbf{v}^- \cdot (\mathbf{i} + \mathbf{a})) \cdot \delta(\mathbf{a} \cdot \mathbf{i}) \end{aligned} \tag{5.7}$$

where $V^-(i,a)$ is the set values with negative feedback given i and a, and $\mathbf{v}^-$ is the embedding of v as a negative value.

5.3.5 Unified AVLEM Framework

With all the components introduced previously, we can learn the embeddings of queries, users, items, aspects and values with a unified framework by maximizing the likelihood of the observed conversations in the training set. For a conversation which was started by user u with an initial request Q_0 and leading to a purchased item i, under the assumptions of multivariate Bernoulli distributions for aspect and values (Section 5.3.4), we need to consider all the aspects both associated with this conversation and not associated. For each aspect that is associated with the conversation, all the values should be taken into account in the generation of both positive and negative values. Let $A(i) = \{a|I(a,v) = 1\} \cup \{a|I(a,v) = -1\}$ be the aspects that appear in the conversation (same as $A(i)$ in Equation (5.4)), and $S_a \setminus A(i)$ be the aspects that have not occurred, where S_a is the set of all the aspects in the collection. Let $T^+_{av} = \{(a, v, S_v \setminus \{v\})|I(a,v) = 1\}$ be the observed instances for positive feedback, and $T^-_{av} = \{(a, v, S_v \setminus \{v\})|I(a,v) = -1\}$ be the observed instances for negative feedback, where S_v is the set of all the possible values in collection and $S_v \setminus \{v\}$ represents all the values that did not co-occur with the corresponding aspect a. The log likelihood of observing the conversation with the reviews of i and u, i.e., R_i and R_u respectively, can be computed as

$$\begin{aligned}\mathcal{L}(R_i, R_u, u, Q_0, &S_a \setminus A(i), T^+_{av}, T^-_{av}, i) \\ &= \log P(R_i, R_u, u, Q_0, S_a \setminus A(i), T^+_{av}, T^-_{av}, i)\end{aligned} \tag{5.8}$$

We assume that the probabilities of R_i, R_u, $S_a \setminus A(i), T^+_{av}, T^-_{av}$ given u, Q_0, i are independent. Words in R_u and R_i are supposed to be generated from the language model of u and i respectively. So R_u is independent from i and Q_0, and R_i is independent

from u and Q_0. We also assume that the positive and negative aspect-value instances, T_{av}^+ and T_{av}^-, only depend on the purchased item i. Initial query intent Q_0 is considered independent from the user preference u. Then Equation (5.8) can be rewritten as:

$$\begin{aligned}
&\mathcal{L}(R_i, R_u, u, Q_0, S_a \setminus A(i), T_{av}^+, T_{av}^-, i) \\
&= \log P(R_i, R_u, S_a \setminus A(i), T_{av}^+, T_{av}^-|u, Q_0, i)P(u, Q_0, i) \\
&= \log \big(P(R_u|u)P(R_i|i) \\
&\qquad P(S_a \setminus A(i)|i)P(T_{av}^+|i)P(T_{av}^-|i)P(i|u, Q_0)P(u)P(Q_0)\big) \\
&\simeq \log P(i|u, Q_0) + \sum_{w \in R_i} \log P(w|i) + \sum_{w \in R_u} \log P(w|u) \\
&\quad + \sum_{a \in S_a \setminus A(i)} \log \big(1 - P(a \in A(i)|i)\big) + \log P(T_{av}^+|i) + \log P(T_{av}^-|i)
\end{aligned} \tag{5.9}$$

$P(u)$ and $P(Q_0)$ are predefined as uniform distributions, and thus ignored in the equation. $P(T_{av}^+|i)$ and $P(T_{av}^-|i)$ can be computed in a similar way. Take $\log P(T_{av}^+|i)$ for instance, we can compute it as:

$$\begin{aligned}
\log P(T_{av}^+|i) &= \sum_{(a,v,\mathcal{V}) \in T_{av}^+} \big(\log P(v, \mathcal{V}|a, i) + \log P(a \in A(i)|i)\big) \\
&= \sum_{(a,v,\mathcal{V}) \in T_{av}^+} \Big(\log P(a \in A(i)|i) + \log P(v \in V^+(a,i)|a,i) \\
&\qquad + \sum_{v' \in \mathcal{V}} \big(1 - P(v' \in V^+(a,i)|a,i)\big)\Big)
\end{aligned} \tag{5.10}$$

where $\mathcal{V} = S_v \setminus \{v\}$ and $V^+(a,i)$ is the set of positive values associated with a and i. $P(T_{av}^-|i)$ can be computed with $V^+(a,i)$ replaced by $V^-(a,i)$, i.e., the set of negative values corresponding to aspect a of i. From Equation (5.9) & (5.10), the overall log-likelihood of an observed conversation is the sum of the log-likelihood for the user language model, item language model, item generation model, aspect generation model, and value generation model.

It is impractical to compute the log likelihood directly since it involves softmax function to compute the probability (Equation (5.3) and (5.1)), which has the sum of a large number of elements as the denominator. Same as [7], we adopt the negative sampling strategy to approximate the estimation of the softmax function. Specifically, β random samples are randomly selected from the corpus according to a predefined distribution and used as negative samples to approximate the denominator of the softmax function. So the log likelihood of the user language model with negative sampling is:

$$\log P(w|u) = \log \delta(\mathbf{u} \cdot \mathbf{w}) + \beta \cdot \mathbb{E}_{w' \sim P_w}[\log \delta(-\mathbf{u} \cdot \mathbf{w}')] \tag{5.11}$$

where P_w is defined as the word distribution in the reviews of the corpus, raised to $\frac{3}{4}$ power [93]. The log likelihood of the item language model can be approximated with u replaced by i in Equation (5.11). Similarly, the log likelihood of the item generation model is computed as:

$$\begin{aligned} \log P(i|u, Q_0) = &\log \delta\Big(\mathbf{i} \cdot \big(\lambda \mathbf{Q_0} + (1-\lambda)\mathbf{u}\big)\Big) \\ &+ \beta \cdot \mathbb{E}_{i' \sim P_i}\Big[\log \delta\Big(-\mathbf{i}' \cdot (\lambda \mathbf{Q_0} + (1-\lambda)\mathbf{u})\Big)\Big] \end{aligned} \tag{5.12}$$

where P_i is predefined as a uniform distribution for items.

Since the sets of aspects and values, namely S_a and S_v, are usually large but the number of aspects and values that appear in a conversation is small, it would be inefficient to consider the whole set of $S_a \backslash A(i)$ and $\mathcal{V}$ (i.e., $S_v \backslash \{v\}$) in Equation (5.9) and (5.10). We random selected β samples from $S_a \backslash A(i)$ and $\mathcal{V}$ to represent the whole set.

The final objective of our model is to optimize the log likelihood of all the conversations in the training set together with L2 regularization to avoid overfitting, i.e.,

$$\begin{aligned}\mathcal{L}' &= \sum_{u,Q_0,i} \mathcal{L}(R_i, R_u, u, Q_0, S_a \setminus A, T_{av}^+, T_{av}^-, i) \\ &+ \gamma\big(\sum_{w\in S_w} \mathbf{w}^2 + \sum_{u\in S_u} \mathbf{u}^2 + \sum_{i\in S_i} \mathbf{i}^2 + \sum_{a\in S_a} \mathbf{a}^2 + \sum_{v\in S_v} (\mathbf{v}^+)^2 + \sum_{v\in S_v} (\mathbf{v}^-)^2\big)\end{aligned} \tag{5.13}$$

where S_u is the set of users, γ is the coefficient for L2 regularization, $\mathbf{v}^+$ and $\mathbf{v}^-$ are the embeddings of v as a positive value and as a negative value respectively, kept in two different lookup tables. All the embeddings are trained simultaneously in our model.

5.3.6 Item Ranking with AVLEM

After we get the embeddings of words, users, items, aspects and values as positive or negative targets, when a user u issues a new query Q_0, in the first iteration, our system ranks an item i based on $P(i|u, Q_0)$ according to Equation (5.1). In the k-th iteration ($k > 1$) of the conversation, besides u and Q_0, the positive and negative feedback on aspect-value pairs collected in previous $k-1$ iterations also act as the basis for ranking. Let AV^+ and AV^- be the aspect-value pairs with positive and negative feedback respectively, item i is ranked according to

$$\begin{aligned}\log P(u, Q_0, AV^+, AV^-|i) &= \log \frac{P(AV^+, AV^-|u, Q_0, i)P(u, Q_0, i)}{P(i)} \\ &= \log \frac{P(AV^+|i)P(AV^-|i)P(i|u, Q_0)P(u)P(Q_0)}{P(i)} \\ &\overset{rank}{=} \sum_{(a,v)\in AV^+} \log\Big(\delta\big(\mathbf{v}^+ \cdot (\mathbf{i}+\mathbf{a})\big) \cdot \delta(\mathbf{a}\cdot\mathbf{i})\Big) \\ &\quad + \sum_{(a,v)\in AV^-} \log\Big(\delta\big(\mathbf{v}^- \cdot (\mathbf{i}+\mathbf{a})\big) \cdot \delta(\mathbf{a}\cdot\mathbf{i})\Big) + \mathbf{i}\cdot\big(\lambda\mathbf{Q_0} + (1-\lambda)\mathbf{u}\big)\end{aligned} \tag{5.14}$$

The time complexity for item ranking is $O(md|S_i|)$, where m is the number of aspect-value pairs used for re-ranking, d is the embedding size, and $|S_i|$ is the total number of items in the corpus.

Table 5.1: Statistics of Amazon datasets.

Dataset	Health & Personal Care	Cell Phones & Accessories	Movies & TV
#Users	38,609	27,879	123,960
#Items	18,534	10,429	50,052
#Reivews	346,355	194,439	1,697,524
#Queries	779	165	248
Query length	8.25±2.16	5.93±1.57	5.31±1.61
#Aspects	1,906	738	6,694
#Values	1,988	1,052	6,297
#AV pairs	15,297	7,111	82,060
#User-query pairs			
Train	231,186	114,177	241,436
Test	282	665	5,209
#Rel items per user-query pair			
Train	1.14±0.48	1.52±1.13	5.40±18.39
Test	1.00±0.00	1.00±0.05	1.10±0.49

5.4 Experimental Setup

In this section, we introduce our experimental settings. We first introduce the dataset and evaluation methodology for our experiments. Then we describe the baseline methods and training settings for our model.

5.4.1 Datasets

Dataset Description. As in previous research on product search [7, 131, 168], we also adopt the Amazon product dataset [89] for experiments. There are millions of customers and products as well as rich meta-data such as reviews, multi-level product categories and product descriptions in the dataset. We used three categories in our experiments, which are *Movies & TV*, *Cell Phones & Accessories* and *Health & Personal Care*. The first one is large-scale while the other two are smaller. We experimented on these datasets to see whether our model is effective on collections of different scales. The statistics of our datasets are shown in Table 5.1. Since there are no datasets that have the sequence of $u \rightarrow Q_0; D_1, \mathcal{Q}_{1,1}, I_{1,1}, \cdots$,

$\mathcal{Q}_{1,m}, I_{1,m} \cdots, D_k, \mathcal{Q}_{k,1}, I_{k,1}, \cdots, \mathcal{Q}_{k,m}, I_{k,m} \rightarrow i$ as a conversation during product search, we need to construct such conversations for the datasets.

Initial Query Construction. To construct initial queries Q_0 in the conversation, we adopt the three-step paradigm of extracting queries for each item, same as the previous work [131, 7, 168]. First, the multi-level category information of each item is extracted from the meta-data. Then, the terms in the categories are concatenated to form a topic string. At last, stopwords and duplicate words are removed. In this way, there can be multiple queries extracted for each item. When a user purchased an item, all the queries associated with the item can be considered as the initial query which is issued by the user that finally leads to purchasing the item. The queries extracted are general and do not reveal specific information about the purchased items. Example queries are "health personal care dietary supplement vitamin", "cell phone accessory international charger", "tv movies" for each category.

Conversation Construction. The essential part to construct a conversation for a user-query pair is to extract the aspect-value pairs from the items. We adopt the aspect-value pair extraction toolkit by Zhang et al. [169, 170] to extract the pairs from the reviews of the items in each dataset. During training, random items were selected as non-relevant results for a user-query pair (u, Q_0) since few items are relevant among the entire collection. Then all the aspect-value pairs extracted from the non-relevant items were used to form corresponding questions. During test time, the aspect-value pairs that were mentioned most in the non-relevant items retrieved in the previous iterations were selected to formulate questions. Table 5.2 shows some common aspect-value pairs extracted from the reviews of an item that corresponds to the example query. In contrast to facets based on which filtering can be applied [82, 132], our extracted aspects and values are more flexible and not exclusive, which makes simple filtering not reasonable. During the conversation, positive or negative feedback on the aspect-value pairs can be constructed.

Table 5.2: Examples of extracted aspect-value pairs.

Query	Aspect	Value
cell phone accessory waterproof case	color	white, black, pink, red
	fit	snug, loose
	material	plastic, rubbery
	plastic	soft, hard, thin, thick
	case	flimsy, protective, sturdy
	cover	dark, clear

Previous works [168, 126] on conversational search and recommendation construct users' response to the system's questions according to their ideal items, which show their hidden intent. In their experiments, the system asks users their preferred values of an aspect, and answers are constructed according to their purchased items or their reviewed restaurants. We also simulate user feedback following the same paradigm. For a question on an aspect-value pair, when the aspect matches an aspect extracted from the purchased item i, if their values also match, the aspect-value pair is considered to have positive feedback, otherwise, the pair is assumed to receive negative feedback. If the aspect in the question does not match any aspect associated with i, no answers are collected from users.

5.4.2 Evaluation Methodology

As in [7], we randomly select 70% of the reviews for each user in the training set and keep the other 30% in the test set. Each review indicates that a user purchases a corresponding item. Then 30% of all the available queries are divided into the test set. If for an item in a training set, all its associated queries are in the test set, we randomly move one query back to the training set. This assures that each item has at least one query in the training data and each tuple of a user, query, purchased item in the test set is not observed in the training set. Finally, all the available user-query pairs in the test set are used to test the performance of the corresponding conversations. Statistics of train/test splits can be found in Table 5.1.

To evaluate the performance of the models in the first k-th iterations in a conversation, we use the freezing ranking paradigm [118, 16], which is commonly used for evaluating relevance feedback, to maintain a rank list. Items shown to the user in the previous $k-1$ iterations are frozen, and the remaining items are re-ranked and appended to the frozen items to form the rank list of all the items. Note that our system does not need to show a long list to the user in each iteration; we keep the items that are not shown in the conversations in the rank lists to avoid that most methods have nearly zero scores for the evaluation metrics. Besides, whenever a relevant item is retrieved in the previous iterations, the ranking of all the items will not be updated in the following iterations. For models that do not utilize feedback, the evaluation is based on the rank lists retrieved with u and Q_0.

We use mean average precision (MAP), mean reciprocal rank (MRR) at cutoff 100, and normalized discounted cumulative gain ($NDCG$) at 10 to evaluate the rank lists in each iteration. MRR indicates the average iterations the system needs to find a relevant item. MAP measures the overall performance of a system in terms of both precision and recall. $NDCG$@10 focuses on the performance of the system to retrieve relevant results in the first 10 iterations, especially in earlier iterations.

5.4.3 Baselines

We compare our aspect-value-based embedding model with three groups of baselines, which are word and embedding based retrieval models that do not consider feedback, and models using item-level negative feedback. Specifically, we have seven representative competing baselines as follows:

- **BM25.** BM25 [116] is a well-known effective retrieval model which scores a document according to a function of the term frequency, inverse document frequency of query terms, and document length. In our experiments, items are

scored according to the BM25 function of their associated reviews in the training set and test queries.

- **QL.** The query likelihood model (QL) [103] is a language model approach for information retrieval which ranks a result according to the log-likelihood that the query words are generated from the unigram language model of the result.

- **LSE.** The latent semantic entity (LSE) model [131] conducts non-personalized product search by representing words and items in the latent entity space. N-grams from the item reviews are extracted and represented with the projection of their word embeddings. Embeddings are learned by maximizing the similarity between the embeddings of items and the n-grams in their reviews. Then queries, which are also n-grams, can be represented with projected word embeddings, and the items are ranked according to their similarities to the query embeddings.

- **HEM.** The hierarchical embedding model (HEM) [7] is the personalized product search model that our model is based on. It has the language models of users and items as well as the item generation model. We use the best version reported in [7] which uses non-linear projected mean for query embeddings and set the query weight $\lambda = 0.5$ (in Equation (5.1)) in both HEM and our own model.

- **Rocchio.** Rocchio [117] is a feedback model in the vector space model that incorporates result-level positive and negative feedback. It refines the query model by bringing it closer to the centroid of relevant results and further from the centroid of non-relevant results. In our task where only non-relevant results are available as the basis for retrieval in the next iteration, Rocchio uses negative feedback alone. We use the BM25 [116] weight for each term in the implementation of Rocchio.

- **SingleNeg.** Karimzadehgan and Zhai [71] proposed this method base on the language model to incorporate negative feedback on results to improve the ranking for the difficult queries. SingleNeg extracts a language model of the negative topic from a batch of non-relevant results by considering they are generated from the mixture of the language model of the negative topic and the background corpus. During re-ranking, the original retrieval score of a result is adjusted with its Kullback-Leibler (KL) divergence with the extracted negative model.

- **MultiNeg.** MultiNeg is the other negative feedback method proposed by Karimzadehgan and Zhai [71]. Different from SingleNeg, it considers each of the negative results is generated from a corresponding negative topic model and uses multiple negative models to adjust the original relevance score.

BM25 and QL are word-based retrieval models. LSE and HEM are embedding-based models for non-personalized and personalized product search. Rocchio, SingleNeg, and MultiNeg incorporate item-level negative feedback collected from previous iterations. For the initial ranking, we use BM25 for Rocchio, QL for SingleNeg and MultiNeg respectively. We get the performance of BM25 and QL using galago[1] with default parameter settings. We implemented Rocchio, SingeNeg and MultiNeg based on galago and tuned the term count for negative model from $\{10, 20, 30, 40, 50\}$, the weight for negative documents from $\{0.01, 0.05, 0.1, 0.2, 0.3, 0.4\}$.

5.4.4 Model Parameter Settings

We implemented our model and HEM with PyTorch [2] and LSE with Tensorflow [3]. LSE, HEM and our model are all trained with stochastic gradient descent for 20

[1] `https://www.lemurproject.org/galago.php`

[2] https://pytorch.org/

[3] https://www.tensorflow.org/

epochs with batch size 64. Initial learning rate is set to 0.5 and gradually decrease to 0 during training. The gradients with global norm larger than 5 were clipped to avoid unstable updates. To reduce the effect of common words, as in [93, 7], we set the sub-sampling rate of words as 10^{-5} for *Cell Phones & Accessories* and *Health & Personal Care*, and 10^{-6} for *Movies & TV*. L2 regularization strength γ was tuned from 0.0 to 0.005. The embedding size d was scanned from $\{100, 200, \cdots, 500\}$. The effect of embedding size will be shown in Section 5.5.3. Negative samples β in Equation (5.11) & (5.7) were set to 5. For conversation construction during training, 2 random items were sampled as non-relevant results, and all the positive and negative values with matched aspects were used in the conversation. For testing, the total number of iterations for retrieval in the conversation was set from 1 to 5. In the first iteration, there is no feedback collected. During each iteration, the number of aspect-value pairs, on which the feedback is provided, namely, m in Equation (5.2), is selected from $\{1, 2, 3\}$. We only report the results of the best settings for all the methods in Section 5.5. [4]

5.5 Results and Discussion

In this section, we discuss the results of our experiments. We first compare the overall retrieval performance of both AVLEM and the state-of-the-art product search baselines in Section 5.5.1. Then we study the effect of different model components, feedback processes, and embedding sizes on each model in the following subsections.

5.5.1 Overall Retrieval Performance

Table 5.3 shows the retrieval performance of all the methods in the conversational product search on different Amazon sub-datasets (i.e., *Movies & TV*, *Cell Phones & Accessories* and *Health & Personal Care*). Specifically, we use BM25 and QL

[4]Our code can be found at `https://github.com/kepingbi/ConvProductSearchNF`

Table 5.3: Comparison between baselines and our model AVLEM. Numbers marked with'*' are the best baseline performance. '+' indicates significant differences between the iterative feedback models and their corresponding initial rankers in Fisher random test [123] with $p < 0.05$, i.e., Rocchio vs BM25, SingleNeg and MultiNeg vs QL, AVLEM$_{pos}$, AVLEM$_{neg}$ and AVLEM$_{all}$ vs AVLEM$_{init}$. '†' denotes significant improvements upon the best baseline. The highest value in each column is in bold.

Dataset	Health & Personal Care			Cell Phones & Accessories			Movies & TV		
Model	MAP	MRR	$NDCG$	MAP	MRR	$NDCG$	MAP	MRR	$NDCG$
BM25	0.055	0.055	0.053	0.065	0.065	0.077	0.012	0.009	0.008
Rocchio	0.055	0.055	0.053	0.065	0.065	0.077	0.012	0.009	0.009^{+}
QL	0.046	0.046	0.048	0.063	0.062	0.076	0.016	0.012	0.015
SingleNeg	0.046	0.046	0.048	0.063	0.062	0.076	0.018^{+}	0.015^{+}	0.017^{+}
MultiNeg	0.046	0.046	0.048	0.063	0.062	0.076	0.018^{+}	0.015^{+}	0.016^{+}
LSE	0.155	0.157	0.195	0.098	0.098	0.084	0.023	0.025	0.027
HEM	0.189*	0.189*	0.201*	0.115*	0.115*	0.116*	0.026*	0.030*	0.030*
AVLEM$_{init}$	$0.227^{\dagger}$	$0.227^{\dagger}$	$0.233^{\dagger}$	$0.126^{\dagger}$	$0.126^{\dagger}$	$0.130^{\dagger}$	$0.028^{\dagger}$	0.030	$0.031^{\dagger}$
AVLEM$_{pos}$	$0.225^{\dagger}$	$0.225^{\dagger}$	$0.250^{+\dagger}$	$0.133^{+\dagger}$	$0.133^{+\dagger}$	$0.135^{+\dagger}$	$0.031^{+\dagger}$	$0.033^{+\dagger}$	$0.035^{+\dagger}$
AVLEM$_{neg}$	$\mathbf{0.260}^{+\dagger}$	$\mathbf{0.260}^{+\dagger}$	$\mathbf{0.305}^{+\dagger}$	$\mathbf{0.154}^{+\dagger}$	$\mathbf{0.154}^{+\dagger}$	$\mathbf{0.177}^{+\dagger}$	$0.033^{+\dagger}$	$0.035^{+\dagger}$	$0.038^{+\dagger}$
AVLEM$_{all}$	$0.236^{+\dagger}$	$0.236^{+\dagger}$	$0.258^{+\dagger}$	$0.145^{+\dagger}$	$0.145^{+\dagger}$	$0.145^{+\dagger}$	$\mathbf{0.034}^{+\dagger}$	$\mathbf{0.036}^{+\dagger}$	$\mathbf{0.042}^{+\dagger}$

as the initial models to generate the first-round retrieval results for Rocchio and SingNeg/MultiNeg, respectively. Also, we refer to the AVLEM without feedback, with positive feedback, with negative feedback, and with both positive and negative feedback on aspect-value pairs as AVLEM$_{init}$, AVLEM$_{pos}$, AVLEM$_{neg}$, and AVLEM$_{all}$, respectively.

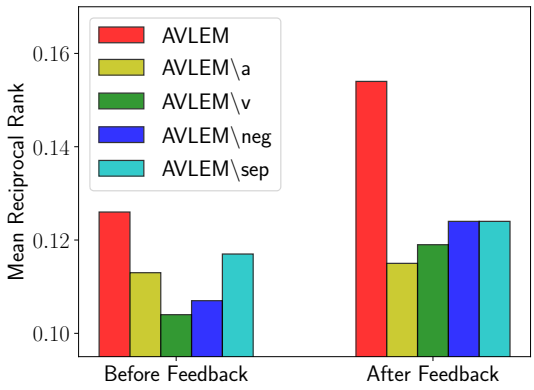
As shown in Table 5.3, term-based retrieval models perform worse than neural embedding models. [5] Without feedback information, QL and BM25 are approximately 50% worse than LSE and HEM on all datasets in our experiments. As discussed by previous studies [131, 7], there are no significant correlations between user purchases and the keyword matching between queries and product reviews. Thus, term-based retrieval models usually produce inferior results in product search. Among different embedding-based product retrieval models, AVLEM$_{init}$ achieves the best performance

[5]MRR and MAP are almost the same for Health & Personal Care and Cell Phones & Accessories since users purchase only 1 item under each query most of the time in these categories (see Table 5.1).

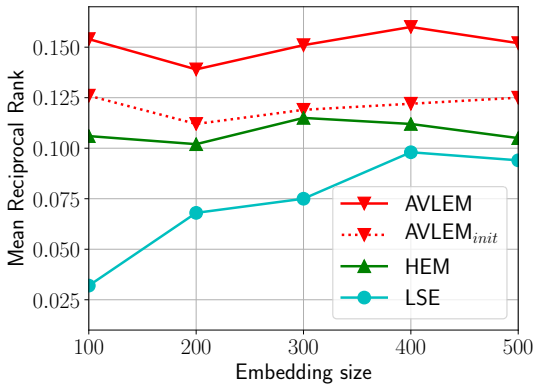
and significantly outperforms HEM and LSE on all three datasets. This indicates that incorporating aspect-value information into search optimizations is generally beneficial for the performance of product search systems.

After a 5-round iterative feedback process, we observe different results for different feedback models. For term-based negative feedback models such as Rocchios, SingleNeg, and MultiNeg, we observe little performance improvement during the feedback process. Comparing to their initial retrieval models in the first iteration (i.e., BM25 and QL), term-based feedback models only achieve significant MRR improvements on *Movies & TV*. For AVLEM, on the other hand, we observe consistent and large improvements over the initial retrieval model (i.e., AVLEM$_{init}$) in all three datasets. The performance of the best AVLEM is approximately 10% to 20% better than AVLEM$_{init}$ in terms of MRR.

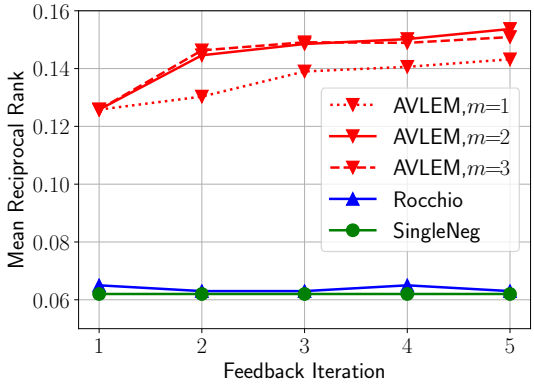
Among different variations of AVLEM, AVLEM$_{neg}$ performs the best on *Cell Phones & Accessories* and *Health & Personal Care*, while AVLEM$_{all}$ performs the best on *Movies & TV*. Overall, it seems that negative aspect-value feedback tends to provide more benefits for AVLEM than positive aspect-value feedback. In a positive feedback scenario, feedback information is "inclusive". In other words, all aspect-value pairs from relevant items could be used to generate positive feedback, but this does not mean that all relevant items should have the same property. For example, a user who tells the system to find a "red" phone case may also be satisfied with a "pink" phone case. In contrast, in a negative feedback scenario, feedback information is "exclusive". When a user says "I don't like red", it means that any items with color "red" is definitely not relevant to this user. Thus, negative feedback information could be more useful for the filtering of irrelevant products.



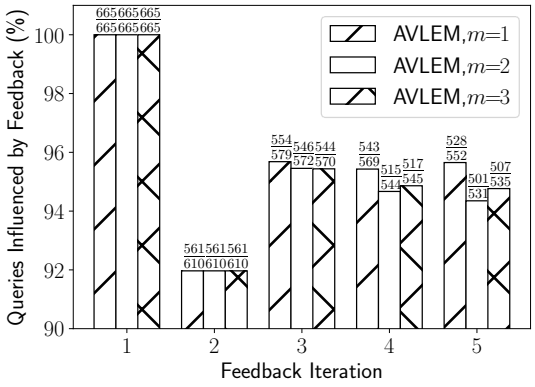
(a) Ablation Study



(b) Effect of Embedding Sizes



(c) Feedback Performance



(d) Feedback Query Coverage

Figure 5.3: The MRR of AVLEM with different components removed and parameter sensitivity analysis of baselines and AVLEM on *Cell Phones&Accessories*.

5.5.2 Ablation Study

In order to evaluate the importance of different model components, we conduct ablation experiments by removing the aspect generation network (i.e., $P(a \in A(i)|i)$ in Equation (5.9) & (5.10)), the value geneartion network (i.e., $P(v \in V^{+/-}(a,i)|a,i)$ in Equation (5.10)), or the negative feedback network (i.e., $P(T_{av}^{-}|i)$ in Equation (5.9)) for AVLEM. We refer them as AVLEM\a, AVLEM\v, and AVLEM\neg, respectively. Also, we refer to the AVLEM that uses a single set of value embedding representations for both $\mathbf{v}^{+}$ and $\mathbf{v}^{-}$ in Equation (5.13) as AVLEM\sep. In AVLEM\neg and AVLEM\sep, we do not have a separate embedding representations for $v \in V^{-}(a,i)$ in $P(v \in V^{-}(a,i)|a,i)$. Instead, we replace $P(v \in V^{-}(a,i)|a,i)$ with $1 - P(v \in V^{+}(a,i)|a,i)$ in Equation (5.13) to train and test these two models.

Figure 5.3a depicts the performance of AVLEM with different components removed on *Cell Phones & Accessories*. We group the results here into two categories – the model performance before feedback (i.e., AVLEM$_{init}$) and the model performance after feedback (i.e., AVLEM). As shown in the figure, removing $P(v \in V^{+/-}(a,i)|a,i)$ in Equation (5.10) (i.e., AVLEM\v) results in a significant drop of retrieval performance for AVLEM before feedback, which means that the relationships between items and aspect-values are important for effectively learning item representations in product search. Also, without the aspect generation model $P(a \in A(i)|i)$, we observe almost no performance improvement on AVLEM\a after the incorporation of feedback information. This indicates that understanding the relationships between items and product aspects is crucial for the use of aspect-value-based feedback signals. Last but not least, we notice that both the removing of $P(T_{av}^{-}|i)$ in Equation (5.9) (i.e., AVLEM\neg) and the unifying of item embeddings in positive and negative feedback (i.e., AVLEM\sep) lead to inferior retrieval performance before and after feedback. As discussed in Section 5.3.4, the use of negative aspect-value pairs and the separate modeling of value embedding in different feedback scenarios are im-

portant for the multivariate Bernoulli assumptions. By replacing $P(v \in V^-(a,i)|a,i)$ with $1 - P(v \in V^+(a,i)|a,i)$, we jeopardize the foundation of AVLEM, which consequentially damages its retrieval performance in our experiments.

5.5.3 Parameter Sensitivity

Effect of Amount of Feedback. Two important hyperparameters control the simulation of conversational feedback in our experiments: the number of feedback iterations and the number of product aspects in each iteration (m). Figure 5.3c depicts the performance of different feedback models with respect to feedback iterations on *Cell Phones & Accessories*. As shown in the figure, the performance of Rocchio and SingleNeg does not show any significant correlations with the increasing of feedback iterations. In contrast, the performance of AVLEM gradually increases when we provide more feedback information. The MRR of AVLEM with 1 product aspect per iteration improves from 0.126 to 0.143 after 5 rounds of feedback. Also, AVLEM generally achieves better performance when we increase the number of feedback aspects from 1 to 3. This indicates that our model can effectively incorporate feedback information in long-term conversations.

To further analyze the effect of multi-iteration feedback, we show the percentage of queries influenced by AVLEM in each iteration on *Cell Phones & Accessories* in Figure 5.3d. Notice that iteration 1 represents the initial retrieval of the feedback process, and this is the reason when all queries are affected by AVLEM. As we can see, the percentages of influenced queries remain roughly unchanged (from 92% to 96%) after each feedback iteration. This means that feedback aspects have been effectively generated by our simulation process in most cases. Also, during the feedback process, the number of available test queries (i.e., the queries with no relevant items retrieved in the previous iterations) gradually decreases from 665 to 531 for the best AVLEM (i.e.,

AVLEM with 2 product aspects per feedback iteration), which means that more and more relevant items have been retrieved.

Effect of Embedding Size. Figure 5.3b shows the sensitivity of both our models and the neural product retrieval baselines (i.e., HEM and LSE) in terms of embedding size on *Cell Phones & Accessories*. While we observe a slight MRR improvement for LSE after increasing the embedding sizes from 100 to 500, we do not see similar patterns for both HEM and AVLEM. Also, the performance gains obtained from the feedback process for our model (AVLEM v.s.AVLEM$_{init}$) are stable with respect to the changes of embedding sizes.

5.6 Summary

To sum up, in this chapter we study negative feedback in product search by breaking it down to fine-grained feedback on aspect-values pairs of results with negative feedback. We propose a conversation paradigm that supports collecting fine-grained feedback during human-system interactions. Then we propose an aspect-value likelihood embedding model (AVLEM) to incorporate both positive and negative feedback on aspect-value pairs. AVLEM consists of the aspect generation model given items and value generation model given items and aspects. One multivariate Bernoulli (MB) distribution is assumed for the aspect generation model, and two other MB distributions are assumed for the generation of positive and negative values. Experiments show that our model significantly outperforms state-of-the-art product search baselines without using feedback and methods using item-level feedback.

CHAPTER 6

ASKING CLARIFYING QUESTIONS BASED ON NEGATIVE FEEDBACK IN INFORMATION-SEEKING CONVERSATIONS

6.1 Introduction

In previous chapters, we have studied how to use positive feedback to identify more relevant answers or an ideal product (Chapter 3 and 4). We also proposed to ask for fine-grained feedback on a product with negative feedback to retrieve the ideal product in fewer iterations (Chapter 5). These scenarios follow a similar conversation paradigm, i.e., the system shows results to users first and then collects their feedback to the previously presented results. If the results receive negative feedback, the system may ask for detailed feedback on the results or the reason why it is non-relevant. There is another typical conversation paradigm for intelligent assistants: asking clarifying questions based on users' initial queries and then showing retrieval results when they are confident about the results to be shown. Some previous work on conversational search in an open domain or on products follows this paradigm [126, 168, 10, 64]. In such systems, user feedback is collected based on clarifying questions instead of previously presented results. In this chapter, we focus on how to leverage feedback on clarifying questions to facilitate information-seeking conversations.

When user queries are ambiguous, faceted, or incomplete, conversational search systems typically ask two types of clarifying questions to users:*special questions* beginning with what/why/how, etc. and *general (yes/no) questions* that can be answered with "yes" or "no". In contrast to special questions that often let a user give specific information about a query, yes/no questions usually require less effort from users since

they provide explicit options and users can easily confirm or deny by saying "yes" or "no". In addition, yes/no clarifying questions make it easier for the system to decide when to show text retrieval results. Users' affirmative answers could enhance the system's confidence in the text retrieval performance. Given these observations, we propose an intent clarification task based on yes/no questions where the system's target is to ask the correct questions about user intent within the fewest conversation turns. When the user intent is confirmed, the system returns the retrieval results.

In the intent clarification task, it is essential to leverage negative feedback about the previously asked questions in the conversation history effectively to select the next question. The principle of using negative feedback is to find a candidate that is dissimilar to the negative results while keeping it relevant to the query. In Web search, documents with negative judgments have limited impact on identifying relevant results due to the large number of potential non-relevant results [139, 140, 71]. In contrast, the intent space of a query is much smaller, providing more opportunity to leverage negative feedback from previous clarifying questions.

In this chapter, we study how to ask clarifying questions based on negative feedback in information-seeking conversations. First, we introduce the intent clarification task based on yes/no questions and its problem formulation. Then we propose a maximum marginal relevance based BERT model (MMR-BERT) to leverage the negative feedback to questions in the conversation history for clarifying question selection. Experimental results show that MMR-BERT significantly outperforms the state-of-the-art baselines in both the intent clarification task and the associated document retrieval task.

6.2 Conversation Intent Clarification

In this section, we first introduce the definition of the conversation intent clarification task. To approach the task, we propose a two-step method to ask clarifying

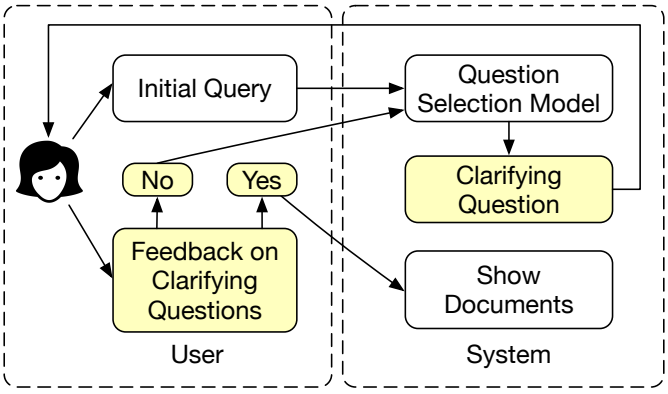


Figure 6.1: A workflow of the intent clarification task.

questions in the conversation. We illustrate the model for initial clarifying question selection in Section 6.2.2 and the model that selects the next question using negative feedback to previous questions in Section 6.2.3.

6.2.1 Task Formulation

Figure 6.1 shows a workflow of the intent clarification task. The user issues an initial query to the system, then the system asks yes/no clarifying questions to the user. When the user provides negative feedback, the system asks another question to confirm the user's intent. Since it is impractical to ask too many questions to users, it is common for conversational search systems to set a limit to the number of asked questions. When the intent is confirmed or the limit of conversation turns is reached, the system returns the document retrieval results. The task is formalized as follows:

Suppose that a user has a specific information need about an ambiguous or faceted topic t. The user issues t as a query to the system. [1] Let $h = ((q_1, a_1), (q_2, a_2), \cdots, (q_{|h|}, a_{|h|}))$ be the conversation history between the user and the system, where the system asks the user $|h|$ clarifying questions $Q_h = \{q_i | 1 \leq i \leq |h|\}$ about the potential intents behind the topic, and the user confirms or denies the corresponding intent indicated

[1] We use topic and query interchangeably in this chapter.

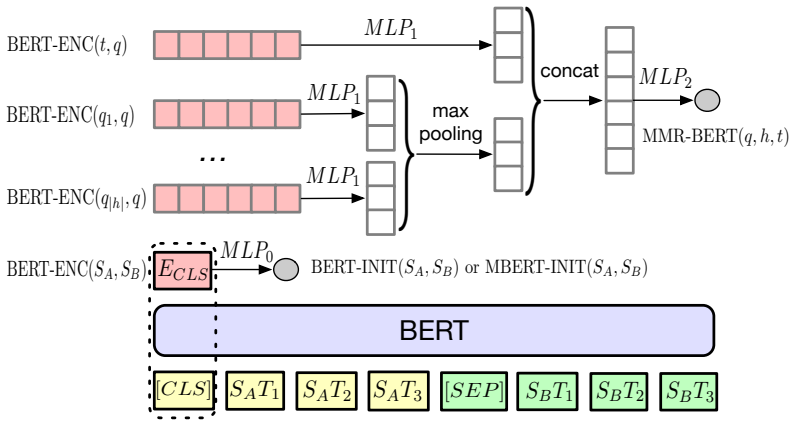


Figure 6.2: Our Maximal Marginal Relevance based BERT Model (MMR-BERT).

in q_i with a_i. For any candidate question q, its label $y(q) = 2$ if it covers the user's true intent, $y(q) = 1$ if it covers other intents of t, and $y(q) = 0$ if it is not relevant to t. The system's target is to identify the user's true intent within the fewest interactions, i.e., $\arg\min(|Q^\star = \{q|y(q) = 2|)$. Since it is not practical to ask too many questions, the system ends the conversation and returns the document retrieval results whenever the user's intent is confirmed or the limit of conversations turns k ($|h| \leq k$) is reached.

6.2.2 First Clarifying Question Selection

The first clarifying question is especially important to elicit user interactions as it will impact the effectiveness of all the future questions and user interactions. The information available to select the initial question is the query itself. Thus it is essential to effectively measure the relevance of a candidate question by how it matches the user query.

Query-question Matching. In recent years, BERT [44] has shown impressive performance in short-text matching tasks by pre-training contextual language models with large external collections and fine-tuning the model based on a local corpus. We leverage BERT to select questions in the intent clarification task. Specifically, we

select the first question based on the relevance score of matching a candidate q to topic t calculated with BERT:

$$s(q,t) = MLP_0(\text{BERT-ENC}(q,t)) \tag{6.1}$$

where BERT-ENC(S_A, S_B) is the output vector of matching sentence A (S_A) and sentence B (S_B) as shown in Figure 6.2, MLP_0 is a multilayer perceptron (MLP) with output dimension 1. Specifically, BERT-ENC(S_A, S_B) inputs the token, segment, and position embeddings of the sequence (*[CLS], tokens in* S_A*, [SEP], tokens in* S_B) to the pre-trained BERT model [44] and take the vector of [CLS] after the transformer encoder layers as output.

Loss Function. We have two ways of calculating the training loss. As a first option, assuming that we do not have any prior knowledge about each user's intent, the retrieval of the first question should simply focus on retrieving questions that are relevant to the initial query string t. Thus we collect a set of query pairs Q^P and each pair consists of a relevant and a non-relevant question, i.e., $Q^P = \{(q^+, q^-)|y(q^+) > 0, y(q^-) = 0\}$. We consider all the questions with positive labels having the same label 1, i.e., $y'(q) = \mathcal{I}(y(q) > 0)$, where $\mathcal{I}$ is an indicator function and equals to 1 when the input condition is true otherwise it is 0. The probability of question q in the entry (pair) E ($E \in Q^P$) being relevant to query topic t is calculated with the softmax function:

$$Prob(y'(q) = 1) = \frac{\exp(s(q,t))}{\sum_{q' \in E} \exp(s(q',t))}, E \in Q^P. \tag{6.2}$$

Then the loss function $\mathcal{L}$ is the cross-entropy between the binary question labels $(1, 0)$ of the pair and the probability distribution of $(Prob(y'(q^+) = 1), Prob(y'(q^-) = 1))$:

$$\mathcal{L}_{\text{BERT-INIT}} = -\sum_{E \in Q^P} \sum_{q \in E} y'(q) \log Prob(y'(q) = 1). \tag{6.3}$$

In this case, the loss function is essentially pairwise loss. We refer to the model trained with Q^P as *BERT-INIT*.

Among the relevant questions of the same query, only questions that match user intents can receive positive feedback and have label 2. As a second option, when we further consider which relevant questions are more likely to receive positive feedback in a prior distribution, the multi-grade label of a question can be used for training.

We extend the set of question pairs Q^P to question triplets $Q^T = \{(q^\bigstar, q^*, q^-)|y(q^\bigstar) = 2, y(q^*) = 1, y(q^- = 0)\}$ and still use the cross-entropy loss to optimize the model. In other words, we train the model according to Equation (6.3) with Q^P replaced by Q^T, i.e.,

$$\mathcal{L}_{\text{MBERT-INIT}} = -\sum_{E \in Q^T} \sum_{q \in E} y(q) \log Prob(y(q) > 0), \tag{6.4}$$

where $Prob(y(q) > 0)$ is calculated based on Equation (6.2) with Q^P replaced by Q^T and E is an entry of triplet, i.e.,

$$Prob(y(q) > 0) = \frac{\exp(s(q,t))}{\sum_{q' \in E} \exp(s(q',t))}, E \in Q^T. \tag{6.5}$$

As in [3], this loss function can be considered as a list-wise loss of the constructed triplets. Since the probability of each question to be a target question is normalized by the scores of all the three questions in the triplet, maximizing the score of question with label 2 will reduce the score of questions with label 1 and 0. Also, questions with larger labels have more impact to the loss. This ensures that the model is optimized to learn higher scores for questions that have larger labels. We refer to this model as *MBERT-INIT*.

6.2.3 Clarifying Intents Using Negative Feedback

While the only basis of the system's decision is topic t in the first conversation turn, the system can refer to conversation history in the following interactions. As

we assume that the system will terminate the conversation and return the documents when the user confirms the question with positive feedback, all the available information for selecting the next clarifying question besides the topic t is negative feedback. It means that the next question should cover a different intent from previous questions while being relevant to topic t.

Inspired by the maximal marginal relevance (MMR) principle in search diversification studies [23], here we propose an MMR-based BERT model (MMR-BERT) to leverage negative feedback in the conversations. In search diversification, the basic idea of MMR is to select the next document by maximizing its relevance to the initial query and dissimilarities to previously selected documents. Similarly, in MMR-BERT, we select the next question by jointly considering the relevance of each candidate question with respect to the initial topic t and their similarities to previous questions. Let Q be the question candidate set, and $Q_h = \{q_i | 1 \leq i \leq |h|\}$ be the set of questions in the conversation history h. Let BERT-ENC(S_A, S_B) be a matching function that takes two pieces of text (i.e., S_A and S_B) as input and outputs an embedding/feature vector to model their similarities. [2] As shown in Figure 6.2, MMR-BERT first obtains the matching of the topic t with candidate question q, i.e., BERT-ENC(t, q) and the matching between each previous question $q_i (1 \leq i \leq |h|)$ and q, i.e., BERT-ENC(q_i, q). Then it maps the obtained vectors to lower d-dimension space ($\mathbb{R}^d$) with a multilayer perceptron (MLP) MLP_1, where each layer is a feed-forward neural network followed by Rectified Linear Unit (ReLU) activation function. The parameters in MLP_1 are shared across multiple matching pairs to let the condensed vectors comparable. Formally, the final matching between x and q is:

[2]Here we use BERT encoder as our matching model because it has been shown to be effective in modeling the latent semantics of text data, which is important for our task since different facets of the same topic often have subtle semantic differences that cannot be captured by simple methods such as keyword matching.

$$
\begin{aligned}
&o(x,q) = MLP_1(\text{BERT-ENC}(x,q)) \in \mathbb{R}^d \\
&x = t \text{ or } q_i, 1 \leq i \leq |h|.
\end{aligned}
\tag{6.6}
$$

The final score of q is computed as:

$$
\text{MMR-BERT}(q,t,h) = MLP_2([o(t,q); MaxPool_{1\leq i\leq |h|} o(q_i,q)]), \tag{6.7}
$$

where $MaxPool$ represents apply max pooling on a group of vectors, $[\cdot;\cdot]$ denotes the concatenation between two vectors, MLP_2 is another MLP for projection to $\mathbb{R}^1$.

Given the user's negative feedback to the asked questions in the conversation history h, the probability of a candidate q covering user intent is calculated according to:

$$
Prob(y(q) = 2|h) = \frac{\exp(\text{MMR-BERT}(q,t,h))}{\sum_{q'\in E} \exp(\text{MMR-BERT}(q',t,h))}, E \in Q^T, \tag{6.8}
$$

where Q^T is a set of triplets, E is a triplet of questions with label 2, 1, and 0, as in Section 6.2.2. To differentiate the questions that would receive positive feedback from users and questions that are relevant to the topic t but do not match user intents, we use the multiple-grade labels in the loss function, as MBERT-INIT in Section 6.2.2. Since $Prob(y(q) = 2, h) = Prob(y(q) = 2|h)Prob(h)$ and $Prob(h)$ is fixed for topic t during training. The loss function can be written as:

$$
\mathcal{L}_{\text{MMR-BERT}} \propto -\sum_{E\in Q^T} \sum_{h\in H(E)} \sum_{q\in E} y(q) \log Prob(y(q) = 2|h), \tag{6.9}
$$

where $H(E)$ is the history set of conversation turns of length 0, 1, 2, and so on, corresponding to triplet entry E. For example, if the questions q_a,q_b, and q_c are already asked for topic t, $H(E) = \{\emptyset, \{q_a\}, \{q_a, q_b\}, \{q_a, q_b, q_c\}\}$. The answers in the history are omitted in the notation since they are all "no". In this way, questions that cover similar intents to historically asked questions Q_h have lower labels than the questions that have target intents and thus will be punished.

Differences from Other BERT-based Models. Most existing BERT-based models for clarifying question selection leverage the topic(query), questions, and answers in the conversation history and do not differentiate answers that are confirmation or denial [10, 64]. In contrast, MMR-BERT is specifically designed to leverage negative feedback from conversation history, which means it uses previously asked questions as input and does not use the answers in the history as they are all denial (we assume that the system would stop asking questions when it has identified the user intent). From the perspective of model design, existing models typically use average BERT representations of each historical conversation turn [10] or concatenate the sequence of a query, question, and answer in each turn as input to BERT models [64]. When used in the intent clarification task, these methods either do not differentiate the effect of each asked question or do not consider the effect of the initial query should be modeled differently from the questions with negative feedback. Following the MMR principle, our MMR-BERT model takes the task characteristics into account and thus can more effectively use negative feedback.

6.3 Experimental Setup

This section introduces the data we use for experiments, how we evaluate the proposed models, the competing methods for comparison, and the technical details in the experiments.

6.3.1 Data

We use Qulac [10] for experiments. As far as we know, it is the only dataset with mostly yes/no clarifying questions in information-seeking conversations. Qulac uses the topics in the TREC Web Track 2009-2012 [29, 30] as initial user queries. These topics are either "ambiguous" or "faceted" and are originally designed for the task of search result diversification. For each topic, Qulac has collected multiple clarifying

questions for each facet (or intent) of the topic through crowd-sourcing; then for each facet of the topic, Qulac obtained the answers to all the questions of the topic from the annotators. The relevance judgments of documents regarding each topic-facet are inherited from the TREC Web track.

We refined Qulac for the intent clarification task by assigning labels 2 or 1 to the questions that receive positive or negative feedback in the answers and label 0 to questions not associated with the topic. Many negative answers in Qulac also include the user's true intent, such as "No. I want to know B." to the question "Do you want to know A?". It is too optimistic to assume users always provide true intents in their answers. Also, in that case, negative feedback does not have difference from positive feedback or is even better. To test how the models performs at incorporating negative feedback alone, we ignore the supplementary information and only keep "no" as user answers. For questions that are not yes/no questions, we consider the answers are negative feedback.

To check whether a model can clarify user intents based on the negative feedback in the conversation history more sufficiently, we enlarge the dataset by including all the questions with label 1 as a 1-turn conversation for each topic-facet. In other words, besides letting the model select the first question, we also enumerate all the questions with label 1 as the first question to check how a model performs under various contexts. The original Qulac enumerates all the questions associated with a query to construct conversations of 1 to 3 turns and only select 1 more question based on the pre-constructed static conversation history. While we also enlarge the data similarly, we only construct conversations with 1 turn, and select questions based on previously selected questions.

The resulting data has 8,962 conversations in total, including 762 conversations of 0-turn (only initial query) and 8,200 1-turn (the added conversations). With the enlarged data, we have many more conversations with various contexts as feedback

Table 6.1: Statistics of our revised version of Qulac.

# topics	198
# faceted/ambiguous topics	141/57
# facets	762
Average/Median facet per topic	3.85±1.05/4
# informational/navigational facets	577/185
# questions/question-answer pairs	2,639/10,277
# *question with positive answers*	*2,007*
Average words per question/answer	9.49±2.53/8.21±4.42
# *expanded conversations*	*8,962*
# conversations starting with 0/1 turns	762/8,200

to test the models and to establish the effectiveness of the results. The statistics are shown in Table 6.1.

6.3.2 Evaluation

We evaluate the models on two tasks: 1) the proposed intent clarification task to see whether it can ask the questions covering the true user intent within fewer conversation turns; 2) the associated document retrieval task to see whether the asked clarifying questions can improve the document retrieval performance. Following [10, 64], we use 5-fold cross-validation for evaluation. We split the topics to each fold according to their id modulo 5. Three folds are used for training, one fold for validation, and one fold for testing. For the question ranking task, we use Query Likelihood (QL) [103] to retrieve an initial set of candidates and conduct re-ranking with BERT-based models. For the document retrieval task, as in [10, 64], we use the revised QL model for retrieval: replacing the original query language model with a convex combination of the language models of the initial query (t) and all the question-answer pairs in the conversation (h).

For the intent clarification task, we *concatenate the question asked in each conversation turn as a ranking list* for evaluation. The primary evaluation metric is MRR calculated based on questions with label 2, which indicates **the number of turns** a model needs to identify true user intent. We also include NDCG@3 and NDCG@5

based on labels 2 and 0 to show how a model identifies the target questions in the first 3 or 5 interactions. To evaluate the overall quality of the clarifying questions, we also use NDCG@3 and NDCG@5 computed using the multi-grade labels 2, 1, and 0 as metrics. These metrics also give rewards to the questions that receive negative feedback from users but are still relevant to the topic. We exclude NDCG@1 since the focus of the evaluation is to see how a model leverages the negative feedback in the context, whereas the first question is ranked based on only the original query. Also, the initial question in most of the conversations is with label 1 in the enlarged dataset regardless of the model used.

For the document retrieval task, we use MRR, Precision(P)@1, NDCG@1, 5, and 20 as the evaluation metrics. MRR measures the position of the first relevant documents. NDCG@1, 5, and 20 indicate the performance based on 5-level labels (0-4) at different positions. Fisher random test [123] with $p < 0.05$ is used to measure statistical significance for both tasks.

6.3.3 Baselines

We include seven representative baselines to select questions and compare their performance to MMR-BERT on both the intent clarification task and the associated document retrieval task:

- **QL**: The Query Likelihood [103] (QL) model is a term-based retrieval model that ranks candidates by the likelihood of generating the query given a candidate, also serving to collect initial candidates.
- **BERT-INIT**: A BERT-based model trained with label 1 and 0 in Section 6.2.2.
- **MBERT-INIT**: A BERT-based model trained with label 2, 1 and 0 as mentioned in Section 6.2.2.

- **SingleNeg**[71]: A negative feedback method that extracts a single negative topic model from the mixture with the language model of background corpus built with the non-relevant results.

- **MMR**: The Maximal Marginal Relevance (MMR) model [23] ranks questions according to the original MMR equation proposed for search diversification as

$$\arg max_{q \in Q \setminus Q_h} \lambda f(t, q) - (1 - \lambda) max_{q' \in Q_h} f(q', q) \tag{6.10}$$

  where λ is a hyper-parameter, and f outputs a similarity score for two pieces of text S_A and S_B. We set $f(S_A, S_B) = sigmoid(\text{BERT-INIT}(S_A, S_B))$.

- **BERT-NeuQS**: BERT-NeuQS [3] uses the *average BERT representations of questions and answers in each historical conversation turn* as well as features from query performance prediction (QPP) for next clarifying question selection. To see the effect of model architecture alone, we did not include the QPP features.

- **BERT-GT**: The Guided Transformer model (BERT-GT) [64] encodes conversation history by inputting *the concatenated sequence of a topic (query), clarifying questions and answers in the history* to a BERT model, guided by top-retrieved questions or documents to select next clarifying question.

QL, BERT-INIT, and MBERT-INIT only use the initial query for ranking while the other models also consider the conversation history. SingleNeg and MMR are based on heuristics. BERT-NeuQS and BERT-GT are state-of-the-art neural models for clarifying question selection. We discard the numbers of other negative feedback methods such as MultiNeg [71] and Rocchio [117] due to their inferior performance. BERT-NeuQS uses the query performance prediction scores of a candidate question for document retrieval to enrich the question representation. Our model significantly

outperforms BERT-NeuQS if we also add this information. However, since we focus on studying which method is better at leveraging the negative feedback, for fair comparisons, we do not include this part for both BERT-NeuQS and our model. BERT-GT works better with questions than documents in our experiments so we only report the setting with questions. MMR-BERT uses the first question from BERT-INIT as its initial question.

6.3.4 Technical Details

We first fine-tuned the "bert-base-uncased" version of BERT [3] using our local documents with 3 epochs. Then we fine-tuned BERT-INIT with 5 epochs allowing all the parameters to be updated. All the other BERT-based models loaded the parameters of the trained BERT-INIT and fixed the parameters in the transformer encoder layers during training. This is because the tremendous amount of parameters in the BERT encoders can easily overwhelm the remaining parameters in different models on the data at Qulac's scale, which makes the model performance unstable. The variance of model performance is huge in multiple runs if we let all the parameters free, which leads to unconvincing comparisons. The limit of conversation turns k was set to 5. We optimized these models with the Adam [76] optimizer and learning rate 0.0005 for 10 epochs. The number of MLP layers that have output dimension 1 was set from $\{1, 2\}$. The dimension of the hidden layer of the 2-layer MLPs was selected from $\{4, 8, 16, 32\}$. λ in Equation (6.10) and the query weight in SingleNeg were scanned from 0.8 to 0.99. Feedback term count in SingleNeg was chosen from $\{10, 20, 30\}$. Top 10 questions were used in BERT-GT. The coefficient to balance the weight of initial query and conversation history in the document retrieval model was scanned from 0 to 1 for each method.

[3] `https://github.com/huggingface/transformers`

Table 6.2: Model performance on intent clarification task evaluated using only label 2 or both label 1 & 2. '*' indicates the best baseline results, and '†' shows the statistically significant improvements over them.

Model	Label 2 only			Label 1&2	
	MRR	NDCG3	NDCG5	NDCG3	NDCG5
QL	0.216	0.130	0.159	0.514	0.565
BERT-INIT	0.235	0.143	0.173	0.531	0.583
MBERT-INIT	0.235	0.144	0.173	0.532*	0.584
SingleNeg	0.217	0.131	0.160	0.513	0.565
MMR	0.237	0.144	0.178	0.531	0.585*
BERT-NeuQS	0.241	0.146	0.182*	0.528	0.580
BERT-GT	0.242*	0.148*	0.178	0.530	0.580
MMR-BERT	**0.248**†	**0.152**†	**0.189**†	**0.533**	**0.586**†

6.4 Results and Discussion

In this section, we show the experimental results of the clarifying question selection task and the associated document retrieval task. We give an analysis of each method's number of success conversations as well as the impact of topic type and facet type in the intent clarification task. At last, we also conduct case studies on the success and failure of our proposed model.

6.4.1 Clarifying Question Selection Results

Overall Performance. As shown in Table 6.2, MMR-BERT has achieved the best performance to identify the target questions that cover true user intents. It outperforms the best baselines significantly regarding almost all the metrics. Note that the evaluation is based on 8,962 conversations and 8,200 of them have the same first negative question in the enlarged data so all the models can refine the question selection only from the second question for most conversations. This limits the improvements of MMR-BERT over the baselines. However, the improvements on about nine thousand data points are significant.

Word-based methods (QL and SingleNeg) are inferior to the other neural methods by a large margin. Also, SingleNeg hardly improves upon QL, indicating that

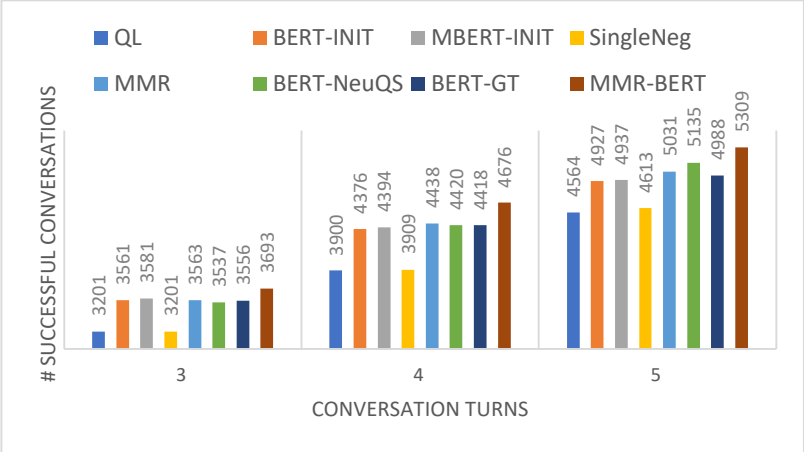


Figure 6.3: Comparison of MMR-BERT and baselines in terms of the cumulative number of success conversations at each turn on the intent clarification task.

word-based topic modeling methods are not effective to incorporate negative feedback in clarifying question selection, probably due to insufficient words to build topic models. The BERT-based methods using the feedback information can identify the first target questions earlier than BERT-INIT and MBERT-INIT. With the similarity function provided by BERT-INIT, MMR can outperform BERT-INIT. The ability of BERT models to measure semantic similarity is essential for the MMR principle to be effective. Moreover, while BERT-NeuQS and BERT-GT improve the metrics regarding label 2, their performance regarding questions with label 1 is harmed. BERT-NeuQS concatenates the topic representation with the average representations of each q-a pair and BERT-GT encode the sequence of the conversation history $(t, (q_1, a_1), \cdots, (q_{|h|}, a_{|h|}))$ as a whole. Thus it could be difficult for them to figure out which part a candidate question should be similar to and which part not. By matching a candidate question with the topic and each historical question individually, MMR-BERT can balance the similarity to the topic and dissimilarity to the historical questions better.

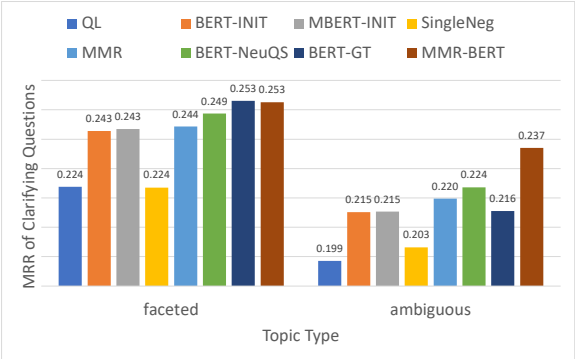
Number of Success Conversations. Figure 6.3 shows the cumulative number of success conversations of each method that correctly identifies user intents at the

third, fourth, and fifth turns. We focus more on how to leverage the negative feedback in the conversation so far rather than how to ask the first clarifying question without feedback information. As shown in the figure, among all the 8,962 conversations, MMR-BERT identifies user intents in 41.2%, 52.2%, and 59.2% conversations by asking at most 3, 4, and 5 clarifying questions. The best baseline at each turn is different while MMR-BERT always has the overall best performance across various turns. This indicates that our MMR-BERT can leverage negative feedback more effectively than the baselines in identifying user intents.

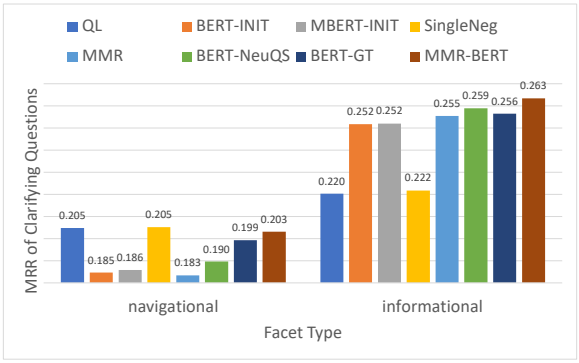
Impact of Topic Type. In Figure 6.4a, we study how MMR-BERT performs on queries of different types compared with other methods. As we mentioned in Section 6.3.1, query topics in Qulac are faceted or ambiguous. An example of a faceted query is "elliptical trainer", which has the facets such as "What are the benefits of an elliptical trainer compared to other fitness machines?", "where can I buy a used or discounted elliptical trainer?", "What are the best elliptical trainers for home use?" and "I'm looking for reviews of elliptical machines." An ambiguous query is a query that has multiple meanings, e.g., "memory", which can refer to human memory, computer memory, and the board game named as memory. From Figure 6.4a, we have two major observations:

1) All the methods perform better on faceted queries than on ambiguous queries. Since QL performs worse on ambiguous queries than on faceted queries by a large margin, the performance of other methods is limited by the quality of initial candidate clarifying questions retrieved by QL. It also indicates that questions for ambiguous queries in the corpus have less word matching than faceted queries.

2) The improvements of MMR-BERT over other methods are much larger on ambiguous queries than on faceted queries. It is essential to differentiate the semantic meanings of various clarifying questions relevant to the same query when leveraging the negative feedback. Clarifying questions of a faceted query are usually about



(a) MRR of each method in the intent clarification task in terms of topic type.



(b) MRR of each method in the intent clarification task in terms of facet type.

subtopics under the small space of the query topic and the words co-occurring with the query in each subtopic have much overlap. Again for the "elliptical trainer" example, the latter associated 3 intents are all related to the purchase need, and the words such as "buy", "best", and "reviews" can co-occur often in the corpus. Thus it is difficult to differentiate these questions even for BERT-based models. In contrast, clarifying questions corresponding to each meaning of an ambiguous query usually consist of different sets of context words, e.g., human memory can have "memory loss" and "brain" in the related texts while computer memory always co-occurs with "disk", "motherboard", etc. As BERT has seen various contexts in a huge corpus during pre-training, they have better capabilities to differentiate the meanings of an ambiguous query compared to the subtopics of a faceted query. However, BERT-NeuQS and BERT-GT cannot fully take advantage of BERT's ability to differentiate semantic meanings due to their architecture, either averaging the representations of historical questions or encoding the sequence of query and the asked questions.

Impact of Facet Type. We compare each method in terms of their performance on different types of intent facets in Figure 6.4b. Similar to the varied performance in terms of topic type, QL performs worse on navigational facets than on informational facets. The clarifying questions that ask about navigational intents sometimes do not match any of the query words such as "are you looking for a specific web site?" and

"any specific company on your mind?" In such cases, the target questions are not included in the candidate pool for re-ranking, which leads to inferior performance on navigational queries.

In addition, we find that neural methods perform worse than word-matching-based methods on navigational queries. Questions that ask about navigational intents are usually in the format of "do you need any specific web page about X (query)?" rather than the typical format of questions about informational intents such as "are you interested in Y (subtopics) of X (query)?" Also, navigational facets are much fewer than informational facets (185 versus 577), which leads to a smaller amount of questions about navigational facets. The supervised neural models tend to promote questions asking about informational intents during re-ranking since they are semantically more similar to the query (talking about their subtopics) and they are more likely to be relevant in the training data. In contrast, word-matching-based methods treat navigational and informational questions similarly since they both hit query words and have similar length. By selecting the next question different from previous questions and relevant to the query, MMR-BERT does not demote questions about navigational facets and does not harm the performance on navigational facets.

6.4.2 Document Retrieval Performance

Table 6.3 and Figure 6.5 show the document retrieval performance of using the original query alone and using the conversations produced by each method. In Table 6.3, we observe that all the question selection methods can promote relevant documents significantly by asking clarifying questions. The questions asked by MMR-BERT achieve the best document retrieval performance, indicating that our model can find users' target information at higher positions by identifying user intents better. Since the model for document retrieval is a simple word-based model, the advantage of asking correct questions may not be reflected in retrieving documents. The cases

Table 6.3: Document retrieval performance with conversations composed by each model. The best baseline results are marked with '*', and the statistically significant improvements over them are marked with'†'.

Model	MMR	P1	NDCG1	NDCG5	NDCG20
OriginalQuery	0.267	0.181	0.121	0.128	0.131
QL	0.292	0.209	0.146	0.142	0.141
BERT-INIT	0.299	0.210*	0.145	0.143	0.143
MBERT-INIT	0.298	0.209	0.143	0.142	0.144*
SingleNeg	0.292	0.209	0.147*	0.142	0.141
MMR	0.301*	0.210*	0.143	0.143	0.144*
BERT-NeuQS	0.296	0.209	0.145	0.145*	0.142
BERT-GT	0.294	0.206	0.141	0.145*	0.143
MMR-BERT	**0.306**†	**0.217**†	**0.151**†	**0.146**	**0.146**†

in Section 6.4.3 show this point. Also, as mentioned in Section 6.3.4, the methods can ask at most 5 questions when they cannot identify user intents. These questions could have more supplementary information than BERT-MMR in finding relevant documents if they are of label 1. Nonetheless, MMR-BERT still achieves significant improvements on 8,962 conversations.

Figure 6.5 confirms the advantage of MMR-BERT by showing that it can retrieve documents relevant to user needs better at earlier turns as well. With more interactions allowed, MMR-BERT can identify more true user intents and thus achieve better document retrieval performance. Among the baselines that select questions using negative feedback, MMR has the best evaluation results most of the time, probably due to its better overall performance in intent clarification, shown in Table 6.2. It boosts questions with label 2 without harming the performance of questions with label 1. Using revised QL for document retrieval, questions of label 1 can also be more helpful than a non-relevant question.

6.4.3 Case Analysis

We extract some representative successful and failure cases of MMR-BERT compared with the best baseline - BERT-GT in terms of MRR in the intent clarification

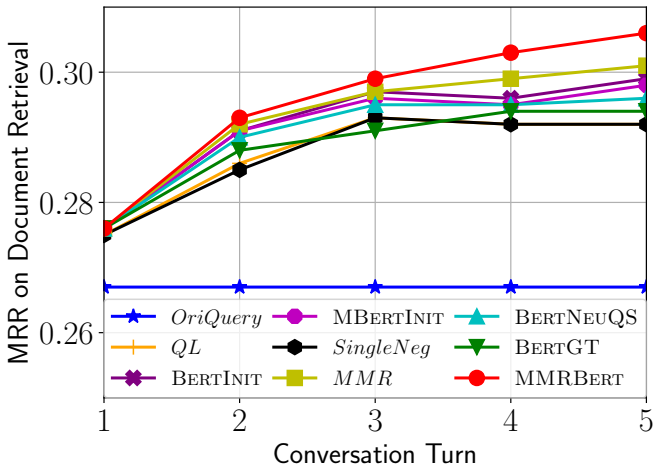


Figure 6.5: MRR at each turn on document retrieval.

task, shown in Table 6.4. We include conversations of faceted and ambiguous queries as well as navigational and informational facets for both good and bad cases to show how the models perform on various types of queries and facets. In these cases, MMR-BERT and BERT-GT have the same initial clarifying questions with negative feedback. These cases show how MMR-BERT and BERT-GT select the next question based on the same previous negative feedback.

Success Cases. MMR-BERT identifies the correct user intent by selecting questions that are relevant to the query while different from previous questions with negative feedback. In contrast, BERT-GT tends to select questions that are similar to both the query and the previous questions. For the example query "diversity", the initial clarifying question asks whether the intent is to find the definition of diversity. MRR-BERT asks the user whether he/she needs the educational materials about diversity in the second turn. However, BERT-GT still asks questions about the definition of diversity twice in the following four turns. For the ambiguous query "flushing", given negative feedback on the first question about toilet flushing, MMR-BERT asks about Flushing in New York in the next question while BERT-GT still asks about the flushing of the same meaning in the second question. For another

Table 6.4: Good and bad cases of MMR-BERT compared with the best baseline - BERT-GT in terms of their MRR differences(ΔMRR(CQ)) in the intent clarification task. The maximal number of conversation turns is 5. ΔMRR(Doc) denotes the MRR difference of the associated document retrieval task after the conversation. Queries are shown in the format of *query(facet description); topic type; facet type*.

"**diversity**"("**How is workplace diversity achieved and managed**?"); **faceted**; **informational**		
BERT-GT	are you looking for a definition of diversity? no would you like the legal definition of diversity? no would you like to know how diversity helps or harms an organization? no do you need the definition of diversity? no would you like to see results about diversity in a business setting? no	ΔMRR(CQ): +0.500 ΔMRR(Doc): +0.667
MMR-BERT	are you looking for a definition of diversity? no are you looking for educational materials about diversity? yes, i need materials on achieving workplace diversity	
"**flushing**"("**Find information about Flushing, a neighborhood in New York City.**"); **ambiguous**; **informational**		
BERT-GT	would you like to see diagrams of how a toilet flushes? no would you like to know about the plumbing mechanisms of flushing? no what aspect of the flushing remonstrance would you like to learn more about? no which battle would you like to learn about how the technique of flushing was used? no what flushing are you interested in toilet or facial? no	ΔMRR(CQ): +0.500 ΔMRR(Doc): +0.005
MMR-BERT	would you like to see diagrams of how a toilet flushes? no are you referring to flushing new york? yes	
"**the sun**"("**Find the homepage for the U.K. newspaper, The Sun.**"); **ambiguous**; **navigational**		
BERT-GT	are you interested in the suns size? no are you interested in objects orbiting the sun? no do you want to know how far the sun is away from different planets? no are you looking for information about how big the sun is? no do you want to know facts about the sun? no	ΔMRR(CQ): +0.500 ΔMRR(Doc): +0.000
MMR-BERT	are you interested in the suns size? no are you looking for the news paper the sun? yes, the uk newspaper	
"**raised gardens**"("**Find photos of raised garden beds.**"); **faceted**; **navigational**		
BERT-GT	do you need information in different types that can be made? no is your request related to raised garden beds? yes, find pictures of it	ΔMRR(CQ): -0.500 ΔMRR(Doc): -0.166
MMR-BERT	do you need information in different types that can be made? no what specific supply would you like to buy for your raised garden? no do you want to take a class about raised gardens? no do you want to buy a book about raised? no do you want to know how to create a raised garden? no	
"**rice**"("**Find recipes for rice, for example fried rice or rice pudding.**"); **ambiguous**; **informational**		
BERT-GT	are you looking for a specific type of rice? no are you looking for recipes that include rice? yes, i want recipes for rice	ΔMRR(CQ): -0.500 ΔMRR(Doc): -0.000
MMR-BERT	are you looking for a specific type of rice? no are you looking for rice university? no do you want to know the nutritional content of rice? no are you referring to a person named rice? no what type of rice dish are you looking? no	
"**flushing**"("**Find a street map of Flushing, NY.**"); **ambiguous**; **navigational**		
BERT-GT	would you like directions to flushing new york? no are you referring to flushing new york? yes, exactly	ΔMRR(CQ): -0.500 ΔMRR(Doc): -0.167
MMR-BERT	would you like directions to flushing new york? no would you like to know about the plumbing mechanisms of flushing? no do you want to know why your face is flushing? no are you looking for a directions to the new york hall of science in flushing meadows corona park? no which battle would you like to learn about how the technique of flushing was used? no	

ambiguous query "the sun", the first clarifying question is about sun size. Based on the negative response, MMR-BERT asks about another meaning of the sun - the newspaper named as the sun. In contrast, the next four questions BERT-GT asks are all about the sun as a star, and the question in the fourth turn is again about the size of the sun. Improvements in identifying the correct clarifying questions can lead to better performance in the associated document retrieval task but it is not always the case probably due to the simplicity of the document retrieval model.

Failure Cases. The questions asked by MMR-BERT in each conversation are more diverse and tend to cover more intents. However, the questions that receive positive feedback sometimes are more semantically similar to the questions with negative feedback than the other questions. In such cases, MMR-BERT fails to identify the correct intents within fewer conversation turns by asking diverse questions. For the faceted query "raised gardens" with intent "find photos of raised garden beds", the initial question does not include any query words, so emphasizing the difference from this question is not helpful and could even be harmful to select next question by introducing noise. For the ambiguous query "rice", the first question asking whether the user wants a specific type of rice receives a negative response. In the following conversations, MMR-BERT asks about other meanings of rice such as Rice University and a person named Rice. BERT-GT selects the question that is also related to the meaning of rice as food in the next turn. Although referring to the same meaning, the aspect of the recipe is the true user intent. Similarly, for the query "flushing", while the user wants the street map of Flushing New York, the question that asks about the direction to Flushing New York receives negative feedback. MMR-BERT selects questions about other meanings of flushing in the next several turns including the mechanism or technique, face flushing, and Flushing meadows corona park. However, the true intent is another facet of the same meaning. These cases argue for other strategies to ask questions such as clarifying meanings for ambiguous queries first

and then asking about the subtopics under the correct meaning. We leave this study as future work. The performance of MMR-BERT in these cases in the associated document retrieval task sometimes is not always worse than BERT-GT, due to some useful information contained in the conversations even though the questions do not receive positive feedback.

6.5 Summary

In this chapter, we study how to ask clarifying questions based on negative feedback in information-seeking conversations. We introduce an intent clarification task based on yes/no clarifying questions, with the goal of asking questions that can uncover the true user intent behind an ambiguous or faced query within the fewest conversation turns. We propose a maximal-marginal-relevance-based BERT model (MMR-BERT) that leverages the negative feedback to the previous questions using the MMR principle. Experimental results on the refined Qulac dataset show that MMR-BERT has significantly better performance than the competing question selection models in both the intent identification task and the associated document retrieval task. Analysis on the impact of topic and facet type shows MMR-BERT outperforms the baselines mainly on ambiguous and navigational queries. Case analysis confirms that MMR-BERT tends to ask diversified questions to users.

Due to the lack of suitable datasets that consist of yes/no questions, we only conducted experiments on Qulac to study intent clarification based on negative feedback. We are aware that there could be different observations for other upcoming datasets in the future. Also, it could be a better policy to generalize to unseen user queries by generating clarifying questions than selecting from pre-collected candidates. We leave collecting other datasets and generating clarifying questions as future work.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this chapter, we briefly summarize our work. We first give an overview of our studies on various aspects of leveraging user feedback and our proposed neural approaches to incorporate each type of feedback. Then we summarize our key conclusions and findings in the previous chapters. At last, we conclude by discussing some potential research directions in studying user feedback.

7.1 Overview of Neural Feedback Approaches in IR

User feedback on the relevance of search results, including documents, answer passages, products, and questions, indicates their information need or preferences. Users' implicit feedback such as clicks is relatively easy to collect and it is also feasible to obtain their explicit feedback during interactionsinmodern scenarios such as voice or text-based conversational search. The feedback can help refine the reranked results to tailor to the user intent. In this dissertation, we explore incorporating positive and negative feedback for ranking effectively. Specifically, we study the following aspects of leveraging feedback:

1. **Iterative Relevance Feedback**. Mobile or voice-based search scenarios argue for iterative feedback with one result in each interaction and passages rather than documents due to severe limitations of space or voice bandwidth. To find more relevant results given known relevant ones from users' positive feedback, we convert conventional relevance feedback (RF) techniques to iterative version and leverage passage-level semantic match to improve the iterative RF performance

on answer passages. The proposed unsupervised iterative RF method is a first step to study positive feedback based on passage embeddings.

2. **Implicit Relevance Feedback**. User clicks indicate their preferences and can be considered as implicit positive feedback. The system can easily collect such data and use them for updating the unshown results when the user requests subsequent pages. As a second step to study positive feedback, we propose a supervised end-to-end neural model to incorporate the user clicks for multi-page product search, which can capture both the semantics of item information and the cooccurrence of clicked and purchased items.

3. **Fine-grained Feedback**. Negative feedback is more valuable than positive feedback since it brings more gains to find a first relevant result than extra relevant results. It is also more challenging as non-relevant results can vary from relevant ones in numerous ways. To gather as much useful information as possible from results with negative feedback, we propose to ask users for fine-grained feedback based on the non-relevant results. We investigate this idea on product search where results are structured so that we can collect feedback on aspect-values of the non-relevant items. We propose an aspect-value likelihood model to incorporate both positive and negative aspect-value-level feedback.

4. **Intent-level Negative Feedback**. In addition to showing search results and asking for relevance judgments on them, another way of collecting feedback is to ask clarifying questions about meanings or subtopics of a query before showing retrieved results. The intent space of a query is usually much smaller than the space of documents or passages, which makes it promising to leverage negative feedback on clarifying questions to reduce the search space and identify user intent. We propose an intent clarification task that aims to ask correct questions about true user intent within the fewest conversation turns. We build our model

on top of the contextual neural language models pretrained with large-scale data, i.e., BERT, to have the ability to differentiate semantic meanings and ask diverse questions based on the negative feedback.

7.2 Summary of Experimental Results

We study each aspect of feedback mentioned in Section 7.1 on various of IR tasks and propose neural models to incorporate the feedback accordingly in the dissertation. The tasks include iterative relevance feedback for answer passage retrieval, leveraging user clicks for multi-page product search, conversational product search based on fine-grained feedback, and intent clarification in information-seeking conversations. We summarize our key findings as follows.

Iterative Relevance Feedback for Answer Passage Retrieval. On standard TREC collections, Robust and Gov2, we observe that iterative RF is at least as effective as standard feedback for document retrieval. On answer passage collections, WebAP [155] and PsgRobust [16], we find that iterative RF is much more powerful than top-k feedback in finding answers. In the IRF experiments, our proposed passage-level semantic match method has produced significant improvements compared to word-based IRF models and other RF models based on term-level semantic similarity. The experiment on retrieval based on one relevant passage shows that in the case where feedback information is scarce passage-level and term-level semantic match are complementary to each other and incorporating both of them leads to even better performance. The effectiveness of our passage embedding-based model indicates that feedback methods do not have to be in the way of extracting expansion terms followed by estimating the term weights and neural approaches are promising.

Leveraging User Clicks for Multi-page Product Search. Experiments on the data collected from Amazon search logs show that the short-term context, i.e., users' implicit feedback in a query session, in multi-page product search significantly

boosts search quality and it is better than the long-term context, i.e., users' global preferences across query sessions. Our supervised end-to-end feedback model that represents user preferences (or feedback topics) with average item embeddings is much more effective than word-based feedback models, again indicating that feedback models do not need to be in the form of query expansion. For a neural retrieval model, the design of a feedback model should combine the representations of results with feedback and the query, where feedback information is incorporated in the latent space without the need of extracting explicit expansion terms.

Conversational Product Search Based on Fine-grained Feedback. On the Amazon Review dataset [89], we extracted aspect-values from product reviews and simulate users' fine-grained feedback based on them. Experimental results show that our proposed aspect-value likelihood model outperforms state-of-the-art product search methods without using feedback and baselines using item-level negative feedback. We also observe that negative feedback at the aspect-value level is more effective than positive feedback in filtering out non-satisfactory items. This is probably because when users provide positive feedback on some aspect-values, they may finally purchase items with other similar attributes. In contrast, when they dislike some aspect-values, they usually do not purchase items with such attributes. Overall, we show that obtaining and using fine-grained feedback on non-relevant results is a promising way to leverage negative feedback.

Intent Clarification in Information-seeking Conversations. We refined the data in an open-domain conversational information-seeking collection, Qulac [10], for the task of asking clarifying questions based on negative feedback. We propose MMR-BERT based on the maximal-marginal-relevance (MMR) principle [23] and pretrained contextual embeddings to leverage negative feedback on clarifying questions. Experiments on the refined Qulac show that MMR-BERT significantly outperforms the state-of-the-art question selection models that are also based on BERT and conven-

tional negative feedback baselines. The questions asked by MMR-BERT also lead to significantly better performance than the competing methods in the associated document retrieval task. Our further analysis indicates that MMR-BERT outperforms the baselines mainly on ambiguous and navigational queries. The ability of pretrained BERT to differentiate semantic meanings is essential for MMR-BERT to effectively use negative feedback.

7.3 Future Work

As interactions between users and retrieval systems have evolved from typing keywords in the search box to conversations, we believe that incorporating user feedback with neural models is a meaningful and important research topic. While we have explored some aspects of the topic towards effective search, there remains some work we are interested in and some challenges we have not discussed yet. Next, we briefly review the directions for future work.

Larger Datasets for Neural Feedback Models. In our studies on user feedback, a big challenge we need to tackle is the lack of data that has dynamic interactions between users and systems. In Chapter 3, we create PsgRobust [1] for iterative feedback on answer passages. In Chapter 4, we collect the data from Amazon search logs. In Chapter 5, we extract aspect-values from item reviews and simulate the conversations by matching the aspect-values of a candidate item and a purchased item. In Chapter 6, we refine Qulac [10] by removing the complementary information in the answers that are negative feedback. Due to the limit of available data to create the suitable collections to study our tasks, sometimes we may not have enough datasets to experiment on and draw more general conclusions (e.g., in Chapter 6). In addition, most datasets in text retrieval only have relevance judgments for a small number of

[1] https://ciir.cs.umass.edu/downloads/PsgRobust/

queries, which limits the exploration of complex neural models with numerous parameters. For instance, BERT has hundreds of millions of parameters, and the small local corpus can only be used to fine-tune the BERT models, which are susceptible to overfitting. A thread of our future work is creating datasets that have more available queries and user interactions. Query logs and Wikipedia topic hierarchies are good resources for us to build datasets upon. On larger datasets, we can do more experiments to verify our findings in the dissertations and analyze the reasons if there are different observations.

Clarifying Question Generation. In this dissertation, we either use templates to construct questions that ask for feedback on passages or aspect-values (Chapter 3, 5) or select questions from a collection of pre-written clarifying questions. To act more intelligently, a system should communicate like a human with diverse sentences. Also, pre-written questions may not fit well when users provide unexpected responses. Thus, it argues for generating questions based on the conversation history in an ideal intelligent retrieval system. The questions should be generated based on aspect-values of a product, meanings of an ambiguous query, subtopics under a meaning, etc., which can be considered as knowledge of the query. To this end, it is essential to construct a knowledge base that captures the structured information of products or the topic hierarchy for queries. We have extracted aspect-values of products in conversational product search while we have not built such a knowledge base for query topics. We can leverage the existing knowledge base, query logs, or relevant documents, to train a model to extract such knowledge for unseen queries and generate questions. Language generation usually requires a large amount of data for training. Due to the expensive effort of creating utterances in various conversations, we can leverage existing questions in other tasks as weak supervision signals to train the generation model.

Questions to Ask for Feedback. In addition to building feedback models to retrieve the next results (passages, products, and questions) given user feedback to previously presented results, which we focus on in this dissertation, it is also very important to study what questions to ask so that the user feedback can provide maximal useful information. The questions can be on the relevance of a passage or document, the preferences of aspect-values of a product, the relevance of meanings or subtopics of a query, etc. There has been some work on selecting feedback documents in conventional top-k feedback based on document relevance, diversity, and density [121, 152, 150, 149]. We aim to study similar topics in the modern search scenarios: in passage retrieval, questions on which passages can benefit reranking quality the most; in product search, questions on what aspect-values can help the system know the user preferences better and identify ideal items as soon as possible; in open-domain information-seeking conversations, from the perspective of document or passage retrieval, what clarifying questions to ask can boost the retrieval performance the most. It could be necessary to build a hierarchy for query topics and ask questions from a higher level to a lower level. Also, query performance prediction techniques can be useful for asking beneficial questions. The models that incorporate feedback and ask questions can be learned simultaneously to complement each other. Furthermore, it is worth studying how to balance question quality and their potential gain since users may quit the session if the system asks unrelated questions.

Implicit Negative Feedback. Users usually skip the results they are not interested in when they browse the result lists. Their skips can be considered as implicit feedback, similar to user clicks, and indicate their negative feedback on the results. In multi-page product search, when users turn to the next page without click any items on the current page, it may indicate that their ideals items are not shown yet. An effective system can leverage this information to refine the ranking of the remaining items. Although such data can be noisy and ground-truth data is hard to collect,

if a user purchases an item in the subsequent pages without clicking any items in the previous pages, it is still a strong signal of negative feedback and the purchased items are the targets. By summarizing the attributes of a batch of non-satisfactory products, it could be promising to identify ideal items by reducing items with similar attributes in the subsequent pages.

Negative Feedback in Free-form User Response. An ideal conversational search assistant should understand user utterances like a human and respond accordingly. In a dialogue, the user response can be free text instead of just a "yes" or "no". It is important to understand what kind of feedback an utterance expresses. For example, given a query "cellphone", the user response "Yes, but better not be iPhone." to the question "Do you like a cellphone that can be unlocked with face recognition?" has negative feedback to "brand: apple" although it confirms "unlock option: face recognition". The system should understand the utterance and extract the corresponding feedback information when searching and filtering items. Another user response "It is a redundant feature" is negative feedback to the feature "unlock option: face recognition". However, the system needs to know that "redundant" means unfavorable to understand this is a negative opinion. Most existing work that leverages conversation history for retrieval does not treat utterances that express a negative opinion specially in the retrieval model [10, 64, 106, 104]. Since term or semantic matching is essential in retrieval models and they usually play a positive role in scoring results, it would lead to user dissatisfaction if utterances with negative feedback are treated the same way. An effective system should understand the opinions delivered in the user utterance and incorporate the negative feedback specially in the retrieval model.

BIBLIOGRAPHY

[1] Aalbersberg, IJsbrand Jan. Incremental relevance feedback. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval* (1992), ACM, pp. 11–22.

[2] Ahuja, Aman, Rao, Nikhil, Katariya, Sumeet, Subbian, Karthik, and Reddy, Chandan K. Language-agnostic representation learning for product search on e-commerce platforms. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (2020), pp. 7–15.

[3] Ai, Qingyao, Bi, Keping, Guo, Jiafeng, and Croft, W Bruce. Learning a deep listwise context model for ranking refinement. *arXiv preprint arXiv:1804.05936* (2018), 135–144.

[4] Ai, Qingyao, Hill, Daniel N, Vishwanathan, SVN, and Croft, W Bruce. A zero attention model for personalized product search. In *CIKM'19* (2019), pp. 379–388.

[5] Ai, Qingyao, Yang, Liu, Guo, Jiafeng, and Croft, W Bruce. Analysis of the paragraph vector model for information retrieval. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval* (2016), ACM, pp. 133–142.

[6] Ai, Qingyao, Yang, Liu, Guo, Jiafeng, and Croft, W Bruce. Improving language estimation with the paragraph vector model for ad-hoc retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016), ACM, pp. 869–872.

[7] Ai, Qingyao, Zhang, Yongfeng, Bi, Keping, Chen, Xu, and Croft, W Bruce. Learning a hierarchical embedding model for personalized product search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2017), ACM, pp. 645–654.

[8] Ai, Qingyao, Zhang, Yongfeng, Bi, Keping, and Croft, W Bruce. Explainable product search with a dynamic relation embedding model. *ACM Transactions on Information Systems (TOIS) 38*, 1 (2019), 1–29.

[9] Aliannejadi, Mohammad, Kiseleva, Julia, Chuklin, Aleksandr, Dalton, Jeff, and Burtsev, Mikhail. Convai3: Generating clarifying questions for open-domain dialogue systems (clariq). *arXiv preprint arXiv:2009.11352* (2020).

[10] Aliannejadi, Mohammad, Zamani, Hamed, Crestani, Fabio, and Croft, W Bruce. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2019), ACM, pp. 475–484.

[11] Allan, James. Relevance feedback with too much data. In *SIGIR* (1995), vol. 95, Citeseer, pp. 337–343.

[12] Allan, James. Incremental relevance feedback for information filtering. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval* (1996), ACM, pp. 270–278.

[13] Allan, James. Hard track overview in trec 2003 high accuracy retrieval from documents. Tech. rep., MASSACHUSETTS UNIV AMHERST CENTER FOR INTELLIGENT INFORMATION RETRIEVAL, 2005.

[14] Belkin, Nicholas J, Cool, Colleen, Stein, Adelheit, and Thiel, Ulrich. Cases, scripts, and information-seeking strategies: On the design of interactive information retrieval systems. *Expert systems with applications 9*, 3 (1995), 379–395.

[15] Bendersky, Michael, Metzler, Donald, and Croft, W Bruce. Learning concept importance using a weighted dependence model. In *Proceedings of the third ACM international conference on Web search and data mining* (2010), pp. 31–40.

[16] Bi, Keping, Ai, Qingyao, and Croft, W Bruce. Iterative relevance feedback for answer passage retrieval with passage-level semantic match. 558–572.

[17] Bi, Keping, Ai, Qingyao, and Croft, W Bruce. Learning a fine-grained review-based transformer model for personalized product search. *arXiv preprint arXiv:2004.09424* (2020).

[18] Bi, Keping, Ai, Qingyao, and Croft, W Bruce. A transformer-based embedding model for personalized product search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020), pp. 1521–1524.

[19] Bi, Keping, Teo, Choon Hui, Dattatreya, Yesh, Mohan, Vijai, and Croft, W Bruce. A study of context dependencies in multi-page product search. In *CIKM'19* (2019), ACM.

[20] Brondwine, Elinor, Shtok, Anna, and Kurland, Oren. Utilizing focused relevance feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016), ACM, pp. 1061–1064.

[21] Brown, Tom B, Mann, Benjamin, Ryder, Nick, Subbiah, Melanie, Kaplan, Jared, Dhariwal, Prafulla, Neelakantan, Arvind, Shyam, Pranav, Sastry, Girish, Askell, Amanda, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

[22] Can, Ethem F, Croft, W Bruce, and Manmatha, R. Incorporating query-specific feedback into learning-to-rank models. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval* (2014), ACM, pp. 1035–1038.

[23] Carbonell, Jaime, and Goldstein, Jade. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (1998), pp. 335–336.

[24] Chen, Minmin. Efficient vector representation for documents through corruption. *arXiv preprint arXiv:1707.02377* (2017).

[25] Cho, Woon Sang, Zhang, Yizhe, Rao, Sudha, Brockett, Chris, and Lee, Sungjin. Generating a common question from multiple documents using multi-source encoder-decoder models. *arXiv preprint arXiv:1910.11483* (2019).

[26] Choi, Eunsol, He, He, Iyyer, Mohit, Yatskar, Mark, Yih, Wen-tau, Choi, Yejin, Liang, Percy, and Zettlemoyer, Luke. Quac: Question answering in context. *arXiv preprint arXiv:1808.07036* (2018).

[27] Christakopoulou, Konstantina, Radlinski, Filip, and Hofmann, Katja. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2016), ACM, pp. 815–824.

[28] Cirillo, C, Chang, Y, and Razon, J. Evaluation of feedback retrieval using modified freezing, residual collection, and test and control groups.

[29] Clarke, Charles L, Craswell, Nick, and Soboroff, Ian. Overview of the trec 2009 web track. Tech. rep., WATERLOO UNIV (ONTARIO), 2009.

[30] Clarke, Charles L, Craswell, Nick, and Voorhees, Ellen M. Overview of the trec 2012 web track. Tech. rep., NATIONAL INST OF STANDARDS AND TECHNOLOGY GAITHERSBURG MD, 2012.

[31] Clarke, Charles LA, Cormack, Gordon V, and Lynam, Thomas R. Exploiting redundancy in question answering. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (2001), pp. 358–365.

[32] Clarke, Charles LA, Cormack, Gordon V, Lynam, Thomas R, Li, CM, and McLearn, Greg L. Web reinforced question answering (multitext experiments for trec 2001).

[33] Cohen, Daniel, and Croft, W Bruce. A hybrid embedding approach to noisy answer passage retrieval. In *European Conference on Information Retrieval* (2018), Springer, pp. 127–140.

[34] Corrada-Emmanuel, Andres, Croft, W Bruce, and Murdock, Vanessa. Answer passage retrieval for question answering. *Tech. Rep.* (2003).

[35] Cox, Ingemar J, Miller, Matt L, Minka, Thomas P, Papathomas, Thomas V, and Yianilos, Peter N. The bayesian image retrieval system, pichunter: theory, implementation, and psychophysical experiments. *IEEE transactions on image processing 9*, 1 (2000), 20–37.

[36] Crestani, Fabio. Learning strategies for an adaptive information retrieval system using neural networks. In *IEEE International Conference on Neural Networks* (1993), IEEE, pp. 244–249.

[37] Crestani, Fabio. Comparing neural and probabilistic relevance feedback in an interactive information retrieval system. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)* (1994), vol. 5, IEEE, pp. 3426–3430.

[38] Croft, W Bruce, Metzler, Donald, and Strohman, Trevor. *Search engines: Information retrieval in practice*, vol. 283. Addison-Wesley Reading, 2010.

[39] Croft, W Bruce, and Thompson, Roger H. I3r: A new approach to the design of document retrieval systems. *Journal of the american society for information science 38*, 6 (1987), 389–404.

[40] Dai, Andrew M., Olah, Christopher, and Le, Quoc V. Document embedding with paragraph vectors. In *NIPS Deep Learning Workshop* (2015).

[41] Dai, Zhuyun, and Callan, Jamie. Deeper text understanding for ir with contextual neural language modeling. In *SIGIR'19* (2019), pp. 985–988.

[42] Dai, Zhuyun, Xiong, Chenyan, Callan, Jamie, and Liu, Zhiyuan. 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* (2018), pp. 126–134.

[43] Dehghani, Mostafa, Azarbonyad, Hosein, Kamps, Jaap, Hiemstra, Djoerd, and Marx, Maarten. Luhn revisited: Significant words language models. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (2016), ACM, pp. 1301–1310.

[44] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. 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)* (2019), pp. 4171–4186.

[45] Di, Wei, Bhardwaj, Anurag, Jagadeesh, Vignesh, Piramuthu, Robinson, and Churchill, Elizabeth. When relevance is not enough: Promoting visual attractiveness for fashion e-commerce. *arXiv preprint arXiv:1406.3561* (2014).

[46] Diaz, Fernando D. Improving relevance feedback in language modeling with score regularization. In *SIGIR* (2008), pp. 807–808.

[47] Duan, Huizhong, and Zhai, ChengXiang. Mining coordinated intent representation for entity search and recommendation. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management* (2015), pp. 333–342.

[48] Duan, Huizhong, Zhai, ChengXiang, Cheng, Jinxing, and Gattani, Abhishek. A probabilistic mixture model for mining and analyzing product search log. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management* (2013), ACM, pp. 2179–2188.

[49] Dumais, Susan, Banko, Michele, Brill, Eric, Lin, Jimmy, and Ng, Andrew. Web question answering: Is more always better? In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval* (2002), pp. 291–298.

[50] Fan, Yixing, Guo, Jiafeng, Lan, Yanyan, Xu, Jun, Zhai, Chengxiang, and Cheng, Xueqi. Modeling diverse relevance patterns in ad-hoc retrieval. In *The 41st international ACM SIGIR conference on research & development in information retrieval* (2018), pp. 375–384.

[51] Ganguly, Debasis, Roy, Dwaipayan, Mitra, Mandar, and Jones, Gareth JF. Word embedding based generalized language model for information retrieval. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2015), ACM, pp. 795–798.

[52] Gao, Jianfeng, Nie, Jian-Yun, Wu, Guangyuan, and Cao, Guihong. Dependence language model for information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval* (2004), pp. 170–177.

[53] Grossman, Maura R, Cormack, Gordon V, and Roegiest, Adam. Trec 2016 total recall track overview. In *TREC* (2016).

[54] Guo, Jiafeng, Fan, Yixing, Ai, Qingyao, and Croft, W Bruce. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (2016), ACM, pp. 55–64.

[55] Guo, Jiafeng, Fan, Yixing, Ai, Qingyao, and Croft, W Bruce. Semantic matching by non-linear word transportation for information retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (2016), ACM, pp. 701–710.

[56] Guo, Jiafeng, Fan, Yixing, Pang, Liang, Yang, Liu, Ai, Qingyao, Zamani, Hamed, Wu, Chen, Croft, W Bruce, and Cheng, Xueqi. A deep look into neural ranking models for information retrieval. *Information Processing & Management 57*, 6 (2020), 102067.

[57] Guo, Yangyang, Cheng, Zhiyong, Nie, Liqiang, Wang, Yinglong, Ma, Jun, and Kankanhalli, Mohan. Attentive long short-term preference modeling for personalized product search. *TOIS 37*, 2 (2019), 1–27.

[58] Guo, Yangyang, Cheng, Zhiyong, Nie, Liqiang, Xu, Xin-Shun, and Kankanhalli, Mohan. Multi-modal preference modeling for product search. In *2018 ACM Multimedia Conference on Multimedia Conference* (2018), ACM, pp. 1865–1873.

[59] Habernal, Ivan, Sukhareva, Maria, Raiber, Fiana, Shtok, Anna, Kurland, Oren, Ronen, Hadar, Bar-Ilan, Judit, and Gurevych, Iryna. New collection announcement: Focused retrieval over the web. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval* (2016), ACM, pp. 701–704.

[60] Harman, Donna. Towards interactive query expansion. In *Proceedings of the 11th annual international ACM SIGIR conference on Research and development in information retrieval* (1988), pp. 321–331.

[61] Harman, Donna. Relevance feedback revisited. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval* (1992), ACM, pp. 1–10.

[62] Harper, David J, and Van Rijsbergen, Cornelis Joost. An evaluation of feedback in document retrieval using co-occurrence data. *Journal of documentation* (1978).

[63] Harter, Stephen P. A probabilistic approach to automatic keyword indexing. part i. on the distribution of specialty words in a technical literature. *Journal of the american society for information science 26*, 4 (1975), 197–206.

[64] Hashemi, Helia, Zamani, Hamed, and Croft, W Bruce. Guided transformer: Leveraging multiple external sources for representation learning in conversational search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020), pp. 1131–1140.

[65] Huang, Po-Sen, He, Xiaodong, Gao, Jianfeng, Deng, Li, Acero, Alex, and Heck, Larry. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management* (2013), pp. 2333–2338.

[66] Huston, Samuel, and Croft, W Bruce. A comparison of retrieval models using term dependencies. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (2014), ACM, pp. 111–120.

[67] Ide, Eleanor. New experiments in relevance feedback. *The SMART retrieval system: Experiments in automatic document processing* (1971), 337–354.

[68] Iwayama, Makoto. Relevance feedback with a small number of relevance judgements: incremental relevance feedback vs. document clustering. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* (2000), ACM, pp. 10–16.

[69] Jin, Xiaoran, Sloan, Marc, and Wang, Jun. Interactive exploratory search for multi page search results. In *Proceedings of the 22nd international conference on World Wide Web* (2013), ACM, pp. 655–666.

[70] Jones, Gareth, Sakai, Tetsuya, Kajiura, Masahiro, and Sumita, Kazuo. Incremental relevance feedback in japanese text retrieval. *Information Retrieval 2*, 4 (2000), 361–384.

[71] Karimzadehgan, Maryam, and Zhai, ChengXiang. Improving retrieval accuracy of difficult queries through generalizing negative document language models. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (2011), ACM, pp. 27–36.

[72] Karmaker Santu, Shubhra Kanti, Sondhi, Parikshit, and Zhai, ChengXiang. On application of learning to rank for e-commerce search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2017), ACM, pp. 475–484.

[73] Keikha, Mostafa, Park, Jae Hyun, and Croft, W Bruce. Evaluating answer passages using summarization measures. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval* (2014), ACM, pp. 963–966.

[74] Keikha, Mostafa, Park, Jae Hyun, Croft, W Bruce, and Sanderson, Mark. Retrieving passages and finding answers. In *Proceedings of the 2014 Australasian Document Computing Symposium* (2014), ACM, p. 81.

[75] Kenter, Tom, and de Rijke, Maarten. Attentive memory networks: Efficient machine reading for conversational search. *CAIR. ACM* (2017).

[76] Kingma, Diederik P, and Ba, Jimmy Lei. Adam: Amethod for stochastic optimization. In *Proc. 3rd Int. Conf. Learn. Representations* (2014).

[77] Krovetz, Robert. Viewing morphology as an inference process. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval* (1993), ACM, pp. 191–202.

[78] Lavrenko, Victor. *A Generative Theory of Relevance*. PhD thesis, 2004. AAI3152722.

[79] Lavrenko, Victor, and Croft, W Bruce. Relevance-based language models. In *ACM SIGIR Forum* (2017), vol. 51, ACM, pp. 260–267.

[80] Le, Quoc, and Mikolov, Tomas. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)* (2014), pp. 1188–1196.

[81] Li, Canjia, Sun, Yingfei, He, Ben, Wang, Le, Hui, Kai, Yates, Andrew, Sun, Le, and Xu, Jungang. Nprf: A neural pseudo relevance feedback framework for ad-hoc information retrieval. *arXiv preprint arXiv:1810.12936* (2018).

[82] Lim, Soon Chong Johnson, Liu, Ying, and Lee, Wing Bun. Multi-facet product information search and retrieval using semantically annotated product family ontology. *Information Processing & Management 46*, 4 (2010), 479–493.

[83] Long, Bo, Bian, Jiang, Dong, Anlei, and Chang, Yi. Enhancing product search by best-selling prediction in e-commerce. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (2012), ACM, pp. 2479–2482.

[84] Lu, Zhengdong, and Li, Hang. A deep architecture for matching short texts. *Advances in neural information processing systems 26* (2013), 1367–1375.

[85] Lv, Sheng-Long, Deng, Zhi-Hong, Yu, Hang, Gao, Ning, and Jiang, Jia-Jian. Fully utilize feedbacks: language model based relevance feedback in information retrieval. In *International Conference on Advanced Data Mining and Applications* (2011), Springer, pp. 395–405.

[86] Lv, Yuanhua, and Zhai, ChengXiang. Adaptive relevance feedback in information retrieval. In *Proceedings of the 18th ACM conference on Information and knowledge management* (2009), ACM, pp. 255–264.

[87] Ma, Yunlong, and Lin, Hongfei. A multiple relevance feedback strategy with positive and negative models. *PloS one 9*, 8 (2014), e104707.

[88] Maron, Melvin Earl, and Kuhns, John L. On relevance, probabilistic indexing and information retrieval. *Journal of the ACM (JACM) 7*, 3 (1960), 216–244.

[89] McAuley, Julian, Pandey, Rahul, and Leskovec, Jure. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2015), ACM, pp. 785–794.

[90] Mcginty, Lorraine, and Smyth, Barry. Adaptive selection: An analysis of critiquing and preference-based feedback in conversational recommender systems. *International Journal of Electronic Commerce 11*, 2 (2006), 35–57.

[91] Metzler, Donald, and Croft, W Bruce. 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* (2005), ACM, pp. 472–479.

[92] Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[93] Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (2013), pp. 3111–3119.

[94] Mogotsi, IC. Christopher d. manning, prabhakar raghavan, and hinrich schütze: Introduction to information retrieval, 2010.

[95] Montazeralghaem, Ali, Zamani, Hamed, and Allan, James. A reinforcement learning framework for relevance feedback. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020), pp. 59–68.

[96] Nogueira, Rodrigo, and Cho, Kyunghyun. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085* (2019).

[97] Oddy, Robert N. Information retrieval through man-machine dialogue. *Journal of documentation 33*, 1 (1977), 1–14.

[98] Pang, Liang, Lan, Yanyan, Guo, Jiafeng, Xu, Jun, Xu, Jingfang, and Cheng, Xueqi. Deeprank: A new deep architecture for relevance ranking in information retrieval. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management* (2017), ACM, pp. 257–266.

[99] Parikh, Nish, and Sundaresan, Neel. Beyond relevance in marketplace search. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (2011), ACM, pp. 2109–2112.

[100] Peltonen, Jaakko, Strahl, Jonathan, and Floréen, Patrik. Negative relevance feedback for exploratory search with visual interactive intent modeling. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces* (2017), ACM, pp. 149–159.

[101] Pennington, Jeffrey, Socher, Richard, and Manning, Christopher D. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (2014), pp. 1532–1543.

[102] Peters, Matthew E, Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, and Zettlemoyer, Luke. Deep contextualized word representations. In *Proceedings of NAACL-HLT* (2018), pp. 2227–2237.

[103] Ponte, Jay M, and Croft, W Bruce. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (1998), ACM, pp. 275–281.

[104] Qu, Chen, Yang, Liu, Chen, Cen, Qiu, Minghui, Croft, W Bruce, and Iyyer, Mohit. Open-retrieval conversational question answering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (2020), pp. 539–548.

[105] Qu, Chen, Yang, Liu, Croft, W Bruce, Trippas, Johanne R, Zhang, Yongfeng, and Qiu, Minghui. Analyzing and characterizing user intent in information-seeking conversations. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (2018), ACM, pp. 989–992.

[106] Qu, Chen, Yang, Liu, Qiu, Minghui, Zhang, Yongfeng, Chen, Cen, Croft, W Bruce, and Iyyer, Mohit. Attentive history selection for conversational question answering. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (2019), pp. 1391–1400.

[107] Rabinovich, Ella, Rom, Ofri, and Kurland, Oren. Utilizing relevance feedback in fusion-based retrieval. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval* (2014), ACM, pp. 313–322.

[108] Radlinski, Filip, and Craswell, Nick. A theoretical framework for conversational search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval* (2017), pp. 117–126.

[109] Rao, Sudha, and Daumé III, Hal. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. *arXiv preprint arXiv:1805.04655* (2018).

[110] Rao, Sudha, and Daumé III, Hal. Answer-based adversarial training for generating clarification questions. *arXiv preprint arXiv:1904.02281* (2019).

[111] Reddy, Siva, Chen, Danqi, and Manning, Christopher D. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics 7* (2019), 249–266.

[112] Rekabsaz, Navid, Lupu, Mihai, Hanbury, Allan, and Zuccon, Guido. Generalizing translation models in the probabilistic relevance framework. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (2016), ACM, pp. 711–720.

[113] Robertson, Stephen E. The probability ranking principle in ir. *Journal of documentation* (1977).

[114] Robertson, Stephen E, and Jones, K Sparck. Relevance weighting of search terms. *Journal of the Association for Information Science and Technology 27*, 3 (1976), 129–146.

[115] Robertson, Stephen E, van Rijsbergen, Cornelis J, and Porter, Martin F. Probabilistic models of indexing and searching. In *SIGIR* (1980), vol. 80, pp. 35–56.

[116] Robertson, Stephen E, Walker, Steve, Jones, Susan, Hancock-Beaulieu, Micheline M, Gatford, Mike, et al. Okapi at trec-3. *Nist Special Publication Sp 109* (1995), 109.

[117] Rocchio, Joseph John. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing* (1971).

[118] Ruthven, Ian, and Lalmas, Mounia. A survey on the use of relevance feedback for information access systems. *The Knowledge Engineering Review 18*, 2 (2003), 95–145.

[119] Salton, Gerard, and Buckley, Chris. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science 41* (1990), 288–297.

[120] Salton, Gerard, Wong, Anita, and Yang, Chung-Shu. A vector space model for automatic indexing. *Communications of the ACM 18*, 11 (1975), 613–620.

[121] Shen, Xuehua, and Zhai, ChengXiang. Active feedback in ad hoc information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval* (2005), ACM, pp. 59–66.

[122] Shen, Yelong, He, Xiaodong, Gao, Jianfeng, Deng, Li, and Mesnil, Grégoire. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web* (2014), pp. 373–374.

[123] Smucker, Mark D, Allan, James, and Carterette, Ben. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management* (2007), ACM, pp. 623–632.

[124] Spina, Damiano, Trippas, Johanne R, Cavedon, Lawrence, and Sanderson, Mark. Extracting audio summaries to support effective spoken document search. *Journal of the Association for Information Science and Technology 68*, 9 (2017), 2101–2115.

[125] Sun, Fei, Guo, Jiafeng, Lan, Yanyan, Xu, Jun, and Cheng, Xueqi. Learning word representations by jointly modeling syntagmatic and paradigmatic relations. In *ACL (1)* (2015), pp. 136–145.

[126] Sun, Yueming, and Zhang, Yi. Conversational recommender system. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval* (2018), SIGIR '18, ACM, pp. 235–244.

[127] Tan, Bin, Velivelli, Atulya, Fang, Hui, and Zhai, ChengXiang. Term feedback for information retrieval with language models. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval* (2007), ACM, pp. 263–270.

[128] Tan, Ming, Santos, Cicero dos, Xiang, Bing, and Zhou, Bowen. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108* (2015).

[129] Taylor, Michael, Guiver, John, Robertson, Stephen, and Minka, Tom. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (2008), ACM, pp. 77–86.

[130] Trippas, Johanne R, Spina, Damiano, Cavedon, Lawrence, Joho, Hideo, and Sanderson, Mark. Informing the design of spoken conversational search: Perspective paper. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval* (2018), pp. 32–41.

[131] Van Gysel, Christophe, de Rijke, Maarten, and Kanoulas, Evangelos. Learning latent vector spaces for product search. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management* (2016), ACM, pp. 165–174.

[132] Vandic, Damir, Frasincar, Flavius, and Kaymak, Uzay. Facet selection algorithms for web product search. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management* (2013), ACM, pp. 2327–2332.

[133] Vassilvitskii, Sergei, and Brill, Eric. Using web-graph distance for relevance feedback in web search. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval* (2006), pp. 147–153.

[134] Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, and Polosukhin, Illia. Attention is all you need. In *Advances in neural information processing systems* (2017), pp. 5998–6008.

[135] Vinay, Vishwa, Wood, Ken, Milic-Frayling, Natasa, and Cox, Ingemar J. Comparing relevance feedback algorithms for web search. In *Special interest tracks and posters of the 14th international conference on World Wide Web* (2005), ACM, pp. 1052–1053.

[136] Vtyurina, Alexandra, Savenkov, Denis, Agichtein, Eugene, and Clarke, Charles LA. Exploring conversational search with humans, assistants, and wizards. In *Proceedings of the 2017 chi conference extended abstracts on human factors in computing systems* (2017), pp. 2187–2193.

[137] Vulić, Ivan, and Moens, Marie-Francine. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2015), ACM, pp. 363–372.

[138] Wang, Daniel, and Nyberg, Eric. A recurrent neural network based answer ranking model for web question answering. In *WebQA Workshop, SIGIR*, vol. 15.

[139] Wang, Xuanhui, Fang, Hui, and Zhai, ChengXiang. Improve retrieval accuracy for difficult queries using negative feedback. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management* (2007), ACM, pp. 991–994.

[140] Wang, Xuanhui, Fang, Hui, and Zhai, ChengXiang. A study of methods for negative relevance feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (2008), ACM, pp. 219–226.

[141] Wang, Yansen, Liu, Chenyi, Huang, Minlie, and Nie, Liqiang. Learning to ask questions in open-domain conversational systems with typed decoders. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics* (2018), pp. 2193–2203.

[142] Wang, Zhenduo, and Ai, Qingyao. Controlling the risk of conversational search via reinforcement learning. *arXiv preprint arXiv:2101.06327* (2021).

[143] Wu, Liang, Hu, Diane, Hong, Liangjie, and Liu, Huan. Turning clicks into purchases: Revenue optimization for product search in e-commerce.

[144] Wu, Zhijing, Mao, Jiaxin, Liu, Yiqun, Zhan, Jingtao, Zheng, Yukun, Zhang, Min, and Ma, Shaoping. Leveraging passage-level cumulative gain for document ranking. In *WWW'20* (2020).

[145] Xia, Fen, Liu, Tie-Yan, Wang, Jue, Zhang, Wensheng, and Li, Hang. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning* (2008), ACM, pp. 1192–1199.

[146] Xiao, Teng, Ren, Jiaxin, Meng, Zaiqiao, Sun, Huan, and Liang, Shangsong. Dynamic bayesian metric learning for personalized product search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (2019), pp. 1693–1702.

[147] Xiong, Chenyan, Dai, Zhuyun, Callan, Jamie, Liu, Zhiyuan, and Power, Russell. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval* (2017), ACM, pp. 55–64.

[148] Xu, Jingjing, Wang, Yuechen, Tang, Duyu, Duan, Nan, Yang, Pengcheng, Zeng, Qi, Zhou, Ming, and Xu, SUN. Asking clarification questions in knowledge-based question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (2019), pp. 1618–1629.

[149] Xu, Zuobing, and Akella, Ram. Active relevance feedback for difficult queries. In *Proceedings of the 17th ACM conference on Information and knowledge management* (2008), ACM, pp. 459–468.

[150] Xu, Zuobing, and Akella, Ram. A bayesian logistic regression model for active relevance feedback. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (2008), ACM, pp. 227–234.

[151] Xu, Zuobing, and Akella, Ram. A new probabilistic retrieval model based on the dirichlet compound multinomial distribution. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (2008), ACM, pp. 427–434.

[152] Xu, Zuobing, Akella, Ram, and Zhang, Yi. Incorporating diversity and density in active learning for relevance feedback. In *European Conference on Information Retrieval* (2007), Springer, pp. 246–257.

[153] Yang, Grace Hui, and Soboroff, Ian. Trec 2016 dynamic domain track overview. In *TREC* (2016).

[154] Yang, Liu, Ai, Qingyao, Guo, Jiafeng, and Croft, W Bruce. anmm: Ranking short answer texts with attention-based neural matching model. In *Proceedings of the 25th ACM international on conference on information and knowledge management* (2016), pp. 287–296.

[155] Yang, Liu, Ai, Qingyao, Spina, Damiano, Chen, Ruey-Cheng, Pang, Liang, Croft, W. Bruce, Guo, Jiafeng, and Scholer, Falk. Beyond factoid qa: Effective methods for non-factoid answer sentence retrieval. In *ECIR* (2016).

[156] Yang, Liu, Qiu, Minghui, Qu, Chen, Guo, Jiafeng, Zhang, Yongfeng, Croft, W Bruce, Huang, Jun, and Chen, Haiqing. Response ranking with deep matching networks and external knowledge in information-seeking conversation systems. In *The 41st international acm sigir conference on research & development in information retrieval* (2018), pp. 245–254.

[157] Yang, Zhilin, Dai, Zihang, Yang, Yiming, Carbonell, Jaime, Salakhutdinov, Ruslan, and Le, Quoc V. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237* (2019).

[158] Yu, Clement T, and Salton, Gerard. Precision weightingan effective automatic indexing method. *Journal of the ACM (JACM) 23*, 1 (1976), 76–88.

[159] Yu, Jun, Mohan, Sunil, Putthividhya, Duangmanee Pew, and Wong, Weng-Keen. Latent dirichlet allocation based diversified retrieval for e-commerce search. In *Proceedings of the 7th ACM international conference on Web search and data mining* (2014), ACM, pp. 463–472.

[160] Zagheli, Hossein Rahmatizadeh, Ariannezhad, Mozhdeh, and Shakery, Azadeh. Negative feedback in the language modeling framework for text recommendation. In *European Conference on Information Retrieval* (2017), Springer, pp. 662–668.

[161] Zamani, Hamed, and Croft, W Bruce. Embedding-based query language models. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval* (2016), ACM, pp. 147–156.

[162] Zamani, Hamed, and Croft, W Bruce. Estimating embedding vectors for queries. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval* (2016), ACM, pp. 123–132.

[163] Zamani, Hamed, and Croft, W. Bruce. Relevance-based word embedding. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* (2017), SIGIR '17.

[164] Zamani, Hamed, Dumais, Susan, Craswell, Nick, Bennett, Paul, and Lueck, Gord. Generating clarifying questions for information retrieval. In *Proceedings of The Web Conference 2020* (2020), pp. 418–428.

[165] Zeng, Wei, Xu, Jun, Lan, Yanyan, Guo, Jiafeng, and Cheng, Xueqi. Multi page search with reinforcement learning to rank. In *Proceedings of the 2018 ACM SIGIR International Conference on Theory of Information Retrieval* (2018), ACM, pp. 175–178.

[166] Zhai, Chengxiang, and Lafferty, John. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management* (2001), ACM, pp. 403–410.

[167] Zhai, Chengxiang, and Lafferty, John. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS) 22*, 2 (2004), 179–214.

[168] Zhang, Yongfeng, Chen, Xu, Ai, Qingyao, Yang, Liu, and Croft, W Bruce. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (2018), ACM, pp. 177–186.

[169] Zhang, Yongfeng, Lai, Guokun, Zhang, Min, Zhang, Yi, Liu, Yiqun, and Ma, Shaoping. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval* (2014), ACM, pp. 83–92.

[170] Zhang, Yongfeng, Zhang, Haochen, Zhang, Min, Liu, Yiqun, and Ma, Shaoping. Do users rate or review?: Boost phrase-level sentiment labeling with review-level sentiment classification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval* (2014), ACM, pp. 1027–1030.

[171] Zhang, Yuan, Wang, Dong, and Zhang, Yan. Neural ir meets graph embedding: A ranking model for product search. In *The World Wide Web Conference* (2019), pp. 2390–2400.

[172] Zhao, Xiangyu, Zhang, Liang, Ding, Zhuoye, Xia, Long, Tang, Jiliang, and Yin, Dawei. Recommendations with negative feedback via pairwise deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2018), KDD '18, ACM, pp. 1040–1048.

[173] Zheng, Guoqing, and Callan, Jamie. Learning to reweight terms with distributed representations. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval* (2015), pp. 575–584.