A Neural Approach to Cross-Lingual Information Retrieval

Submitted in partial fulfillment of the requirements for

the degree of

Master of Science

in

Information Networking

Qing Liu

B.S., Computer Science, Beijing University of Posts and Telecommunications

Carnegie Mellon University
Pittsburgh, PA

April, 2018

Acknowledgements

Firstly, I would like to express my sincere gratitude to my thesis advisor Prof. Jamie Callan for guiding me and providing me insightful suggestions during the whole process of this work, for all the time he spent in helping with my research, defense presentation and writing work. I also want to thank him for letting me find my area of interest by providing a wonderful course in search engines. This work is inspired by him from the very beginning.

Also, I want to thank my thesis reader Zhuyun Dai for supporting me and inspiring me with all those marvelous ideas all along the way. The key component of this work, a kernel based neural ranking model is from Zhuyun's work. I want to thank Zhuyun for providing me help and answering my questions whenever I'm in need. This work can't be finished without her help.

Besides my thesis advisor and reader, I want to thank Yukun Zheng, Cheng Luo for providing me helpful information to improve system performance in experiments. Their insights about Chinese segmentation and knowledge about professional tools helped me a lot when building the software.

This work is self-funded and there is no funding or any funder.

Lastly, I want to thank my husband Xiaodu Ding for all his encouragement and support. Thank you for being with me for all those sleepless nights when I was struggling and doubting. Thank you for giving me the courage to work all the way through.

Abstract

With the rapid growth of world-wide information accessibility, cross-language information retrieval (CLIR) has become a prominent concern for search engines. Traditional CLIR technologies require special purpose components and need high quality translation knowledge (e.g. machine readable dictionaries, machine translation systems) and careful tuning to achieve high ranking performance. However, with the help of a neural network architecture, it's possible to solve CLIR problem without extra tuning or special components. This work proposes a bilingual training approach, a neural CLIR solution allowing automatic learning of translation relationships from noisy translation knowledge. External sources of translation knowledge are used to generate bilingual training data then the bilingual training data is fed into a kernel based neural ranking model. During the end-to-end training, word embeddings are tuned to preserve translation relationships between bilingual word pairs and also tailored for the ranking task. In experiments we show that the bilingual training approach outperforms traditional CLIR techniques given the same external translation knowledge source and it's able to yield ranking results as good as that of a monolingual information retrieval system.

In experiments we investigate the source of effectiveness for our neural CLIR approach by analyzing the pattern of trained word embeddings. Also, possible methods to further improve performance are explored in experiments, including cleaning training data by removing ambiguous training queries, exploring whether more training data will improve the performance by learning the relationship between training dataset size and model performance, and investigating the affect of English queries' text-transform in training data. Lastly, we design an experiment that analyzes the quality of testing query translation to quantify the model performance in a real testing scenario where model takes manually written English queries as input.

Table of Contents

List of Tables

List of Figures

Symbols

$target\_query$	A query in target language.
src_query	A query in source language.
$target\_positive\_document$	A document in target language considered as relevant.
$target\_negative\_document$	A document in target language considered as not relevant.
$q = q_1, q_2 ... q_n$	A query consists of query terms q_1,q_2 ...q_n.
$d = d_1, d_2 ... d_m$	A document consists of document terms d_1,d_2 ...d_m.
$\overrightarrow{v_{q_i}}$	A vector of word embedding for query term q_i.
$\overrightarrow{v_{d_i}}$	A vector of word embedding for document term d_i.

Abbreviations

IR	Information retrieval.
CLIR	Cross-lingual information retrieval.
OOV	Out of vocabulary.
POS	Part of speech.
K-NRM	Kernel based neural ranking model.
MT	Machine translation.
ch	Chinese.
en	English.
qry	Query.
doc	Document.

1

Introduction

With the rise of the Internet, data and documents within wold-wide became accessible to people in different countries. People are more and more interested in exploring the contents that are not written in their native language. Information retrieval (IR) task that involves in document and queries in different languages is referred to as cross-lingual information retrieval (CLIR) [1]. On the other hand, a IR task where user tries to retrieve documents written in the same language with query is called mono-lingual information retrieval (MLIR).

The most significant advantage of CLIR over a MLIR is that CLIR is able to provide much more thorough retrieval result. In MLIR, documents retrieved must be in the same language with query, therefore the result is limited in one language. While in CLIR, documents written in languages other than query language can be retrieved. This can be very helpful in scenario where recall rate is valued, like scientific, medical or technical search.

CLIR tasks require translation between different language pairs. The challenge of CLIR lies in how to resolve ambiguity so that high quality translation result that maximizes retrieval accuracy can be obtained. Each term can have multiple choice of translation in another language. Resolving ambiguity refers to the process of choosing best translation candidate. Ambiguity happens at both single term translation and phrase translation. Phrase translation has been shown to be especially problematic [4]. The reason for this is that terms may have different translation when they are used individually and used in phrase, like “deep learning”. Failure to identify or translate phrase properly will result in translation error that have serious affect on retrieval accuracy.

Traditional CLIR approaches spend great effort on mitigating the affect of translation ambiguity. State-of-the-art traditional CLIR system can be nearly as effective as monolingual systems[5]. However, building such system takes non-trivial work. Complex procedure and careful tuning are required in such process. Another limitation in traditional CLIR is that once failed to eliminate translation error, problematic translation will cause damage in retrieval quality seriously. This is because translation results are used directly in retrieval tasks, for example, translate queries into the same language with documents then use translated queries to conduct search on documents. Once problematic translation is given, it’s directly used in retrieval process can cast huge affect on retrieval quality.

In this work, we proposed a neural CLIR approach, which addresses the above two limitations in traditional CLIR works. Traditional CLIR approaches are built on word-based ranking models, like BM25 [20] and Indri [22]. Such models can only consider exact-match between words, like “Apple” equals “Apple”. On the other hand, a neural ranking model can learn soft-match [2] from training data, like “Apple” is related to “iPhone”, “iMac”. Soft-Match allows a neural ranking model to capture and use a lot more relevant evidence than a traditional word-based ranking model does. Our work tries to extend the concept of soft-match from a monolingual scenario to bilingual scenario, like “Apple” is related to “苹果” (apple), “手机” (phone). By switching to a neural architecture, now CLIR can be treated as a machine learning problem. This avoids a lot manual tuning process when pursuing a higher performance. Also, it allows the model to be more robust to translation error because now translation results are used in training data instead of queries. The neural ranking model can learn how to use translation results in a way that maximize the ranking accuracy.

A neural CLIR approach is consist of two components: bilingual training data and neural ranking model. Bilingual training data we used is a Chinese, English bilingual search log generated from a monolingual Chinese search log by using machine translation to translated Chinese queries to English. The neural ranking model we used is a kernel-based neural ranking model (K-NRM) [2]. In K-NRM, words are represented with word embeddings and during the end-to-end training of K-NRM word embeddings for both English term and Chinese term can be represented and trained in the same vector space. This allows the word embeddings between a Chinese term and English term be be comparable and their embedding similarity can be used as soft-mach signals. During ranking, learned soft-matches are used to determine the relevance score for a bilingual query, document pair.

Our experiment results show that neural CLIR approach can reach the performance of a monolingual retrieval system. We also compared the performance of traditional CLIR approach on neural ranking model with that of a neural ranking model trained with bilingual training data and found that given the same translation knowledge source, bilingual training data gives better performance than tradition CLIR approach on the same neural ranking model.

In experiments we investigate the source of effectiveness for our neural CLIR approach by analyzing the pattern of trained word embeddings. Also, possible methods to further improve performance are explored in experiments, including cleaning training data by identifying and removing ambiguous training queries, exploring whether more training data will improve the performance by learning the relationship between training dataset size and model performance, and investigating the affect of English queries' text-transform in training data. Lastly, we design an experiment that analyzes the quality of testing query translation to quantify the model performance in a real testing scenario where model takes manually written English queries as input.

The remaining part of which work is organized in the following way. Section 2 introduces related work about traditional CLIR approaches and a newly emerging field, neural ranking, that this work is based on. Section 3 gives a detailed introduction of this work's contribution, a neural CLIR approach. Section 4 introduces the experiment methodologies, including dataset used in experiment, metrics used in results evaluation, retrieval methods as well as translation approaches tested in experiments. Section 5 shows experiment settings and designs for CLIR on traditional retrieval models while section 6 includes experiments about CLIR on neural ranking models. Conclusions from this work are discussed in section 7.

2
Related work

This section gives an introduction about the design choices and approaches of previous work in traditional CLIR. Mechanisms used in traditional CLIR approaches to improve performance is also discussed. Last, newly emerging technologies like word embedding and neural ranking are introduced for their potentiality in providing new CLIR solutions.

2.1 Translation strategy

Mono-lingual Information retrieval tasks rank documents based on their similarity to a specific query. CLIR tasks require capturing such similarity across language boundaries. In CLIR systems, queries might be written in one language while documents are written in another. To measure the similarity between such bilingual query and document pairs, translation from current source language to a target language is required. There are four strategies for translation in CLIR based on whether the query or document is translated: use untranslated query to match untranslated document in another language, translate query to match the language of document, translate document to match the language of query, and translate both query and document into a common representation.

The first one is often used as a lower bound for CLIR system performance for that in a CLIR system without any kind of translation, only terms sharing the same representation in both query language and document language can be matched. Any translation approach should improve that baseline performance because the common vocabulary between query set and document set is enlarged.

The second strategy is often referred to as “query translation” while the third strategy is referred to as “document translation”. The former approach translates the query from current source language to the target language that document is written in at retrieval time while the latter one translates every document in the corpus to the query language at indexing time.

The most significant advantage of query translation method over document translation method is its high computation efficiency and flexibility. Compared with document, queries tend to be very short. Also, the cost of extending a query translation based system to support another language pair is negligible compared with a document translation based system. Query translation module can be added easily to existing IR system. Another advantage is memory efficiency. The space required by document translation is linear to the number of languages involved in the CLIR task, while query translation translate query on the fly and doesn’t require extra space.

However, translation for queries will suffer a penalty in accuracy for the lack of context information. Queries are usually shown as phrase and lacking complete sentence structure, therefore they are more susceptible to translation error like ambiguity. In Oard [11], experiments comparing the performance of document translation and query translation in CLIR tasks have been done on TREC-6 CLIR SDA/NZZ collection, which is a German collection designed specially for CLIR experiment. Experiment results showed that machine-translation based document translation outperformed machine-translation based query translation by 6% in precision for short queries and 40% for long queries. According to the result, document translation works especially well when length of queries are long. Note that the improvement of document translation in long queries is largest is that in this experiment setting, long queries are generally harder than short queries based on the monolingual experiment result.

Except for translation accuracy, another feature that makes document translation attractive is its ability to support interactive multilingual applications. It supports the scenario where user want to browse through retrieval result in the query language. Query translation systems can't support such service as good as the ones based on document translation for that machine translation on large content is very time consuming and users are expecting short response time from IR systems.

Even though document translation has the advantage of better performance gained from higher translation accuracy and convenience of demonstrating retrieved document in target language, it's unrealistic in most of the applications for the computation and space efficiency issues. Especially for web search environment, where the amount of corpus is inestimable and new content is being created every moment. With the rapidly update of indexed content, it's impractical to translate and store all updated contents. Also, it's unrealistic to apply document translation on such large volumes of data. However web search has a strong demand for CLIR capability for its world-wide distribution and transnational nature. Compared with document translation, query translation fits the demand of web search perfectly well. Query translation based methods exhibit high efficiency and flexibility that suits the task well. The intense demand for CLIR in web search made query translation become the most popular translation strategy in recent decades. Therefore, this work will be focus on query translation techniques and document translation won't be further discussed in this work.

2.2 Translation Approach

While translation strategy states the problem of what to translate, translation approach focuses on how to translate. Commonly used translation approaches can be divided into text translation and term vector translation based on the resources used for translation. Text translation refers to the process of translating text from source language to target language with machine translation systems, while term vector translation is defined as the process of mapping each term in source language to all of its definitions in target language, like dictionary-based translation.

2.2.1 Machine translation

Machine translation has been an extensively studied subfield of artificial intelligence and natural language processing. This allows machine translation based IR systems to exploit the result of machine translation related research studies and wide range of commercial products.

With sophisticated machine translation system involved, text translation has several advantages over term vector translation. The first advantage is that machine translation resolves ambiguity directly during translation by analyzing structures in source and target language. Another advantage is that machine translation is capable of translating words that are not included in the dictionary and such translation is called transliteration [13]. For example, rule-based machine translation first detects the semantic and syntactic structures of source language and convert them to corresponding structures in target language. After aligning the source and target language, terms can be mapped directly. However, machine translation can be impractical for large collections for that it's generally computationally expensive [14].

Another concern when using machine translation for CLIR is that in query translation the text used for translation is usually short. As discussed before, machine translation has been proven to be effective when applied on document translation. However, due to the requirement in short response time and limitation in computation/space resources, query translation is preferred in most of the scenarios. Compared with documents, queries are short and usually can't form complete sentences. Structure detection and context analysis is typically more difficult on queries. As a result, the capacity of machine translation based approaches is weaken by the limited length of queries. According to the experiment result in Oard [11], the performance of machine-translation based query translation lies between that of simple dictionary based approach and dictionary based approach with linguistic processing. For long queries, machine-translation based query translation outperforms dictionary based approaches. This result implies the possibility of improving long query performance by using machine translation techniques.

2.2.2 Dictionary based translation

As for term vector translation, dictionary based translation is the most commonly used approach.Dictionary based translation is a special translation approach that mainly used in IR task. Unlike machine translation which gives the final multi-word translation result, dictionary based translation gives a list of translation candidates terms for each term being translated. This suits the IR task well because IR systems use a bag-of-words representation of documents and word sequences are not considered if not specified in query. Dictionary based translation requires a bilingual dictionary in machine readable format. As the source of translation capacity, quality of bilingual dictionary largely determines the performance of CLIR system.

Obtain bilingual dictionary The first issue is how to obtain a bilingual dictionary that specifically suits IR task. Bilingual dictionaries in machine readable format are more and more accessible. But such dictionaries are still designed for human readers and tend to contain a lot of contextual words that are not suitable for IR task. This type of dictionary is called “bilingual general language dictionaries”. Bilingual general language dictionaries is consist of different fields like pronunciation, examples, definitions or encyclopedic definitions.The only information CLIR tasks need is the the direct translation for each term. Such bilingual dictionary that contains a list of equivalent translations for each source language term is defined as “bilingual thesaurus”. In this work, if there is no specific notation, “bilingual dictionary” is referring to “bilingual thesaurus”.

Previous work shows that an automatic filtering process can be used in generating bilingual thesaurus from general language dictionary [1]. First, fields that not containing direct translation information was filtered out, like pronunciation, example and etymology. The fields used for translation extraction are those contains definitions. Then, stop words and repeat words are removed and the left words are used as translation terms. Such simple automatic filtering results in a noisy set of translation term. For example, French term “radiation” has English translation terms: disbarring, expulsion, radiation, striking, register, loss, license, practice, medicine. An ideal translation term set picked manually should only contains radiation, expulsion and disbarring. In this example, noisy translation words like “medicine”, “license” will lead to a twisted result in retrieval tasks. Such noisy translation set is considered as a source of error in CLIR process. From the experiment result in Hull [1], a system based on automatically built dictionary can only reach 60% of average precision of the monolingual baseline. Although there is a significant gap in performance between queries translated from automatically generated dictionary and monolingual baseline, other experiments shows that there can be dramatic improvement in performance with certain improvement on the dictionary. Bases on experiment result, with noisy translation terms manually removed, the performance can be improved to 68%; with phrase translation capacity added to the dictionary, the performance can be improved to 90%. This result of Hull [1] indicates that dictionary based query translation is a feasible approach for CLIR task and that the performance of built CLIR system is highly depended on the quality of the bilingual dictionary used. With an ideal bilingual dictionary (noise-free translation terms, multi-word phrase are correctly handled), CLIR system can be almost as good as a monolingual system.

Resources other than general bilingual dictionaries can be used to generate bilingual thesaurus. The simple direct translation filtering strategy described in Hull [1] can be further extended with more complex rules and filtering can be done on document collections in target language, which is more domain related. A "hypernym-hyponym extraction" on Japanese and US patents was proposed in Nanba [15] to automatically constructing Japanese-US bilingual thesaurus. Such method relies on detection of patterns like "A such as B" to identify a hypernym-hyponym pair "A" and "B". Extracted hypernym-hyponym pairs in English and Japanese are aligned with citation analysis techniques and finally construct a bilingual thesaurus.

Reduce translation error for dictionary based translation To achieve a better performance in CLIR task, it's important to first figure out the source of translation error. Previous work shows that the translation error in CLIR process is mainly caused by three factors [4].

Adding extraneous terms to the query The first is adding extraneous terms to the query. Irrelevant terms may be introduced in translation process due to noisy translation set, like described in previous section. Such error can be reduced by a large extent with a carefully constructed bilingual dictionary. While another source of extraneous terms is that common terms usually have multiple definitions. This type of error can't be eliminated during dictionary construction phase for that term translation is ambiguous naturally: terms may have different senses in different context. For terms with multiple candidate translation terms, several selection strategies have been explored by previous work.

The most straight forward strategy is simply keeping every candidate (every word strategy). Every word strategy has been more commonly implemented. Studies has been done on how to cut down on number of translations selected to improve translation quality. Filtering strategy based on part of speech (POS) analysis has been used to reduce ambiguity [4]. This method exploits the POS tag for each possible translation provide by the bilingual dictionary. Queries in source language are first tagged with a part of speech tagger and when selecting translation candidates, only the candidate with the same POS tag will be selected. Experiment results shows that translation using POS to disambiguate outperforms word-by-word translation without disambiguation attempt by 22% in terms of average precision.

Another intuitive strategy is to pick a single word when there are several candidates available (single word strategy). A simple random selection approach can be used for this purpose. Experiments in Oard [11] compared the every word strategy and random selection strategy. Result indicates that a random single word strategy is no worse than every word strategy: random single word approach can reach 98% of the performance of every word approach. This result points out that seeking improvement over arbitrary choice may be as meaningful as seeking efficient ways to cut down number of translations selected.

External resources other than bilingual dictionary can be helpful when trying to select the best candidate. Parallel corpus can be used to support disambiguation methods [4]. Parallel corpus is defined as a set of documents and their translations in one or more other languages. The disambiguation is proceeded in the following way: first, retrieve top 30 documents by running original query on source language corpus. Then corresponding 30 documents in parallel target language set are collected and top 5,000 terms are scored based on Rocchio ranking [23]. All candidate translation terms are ranked by their scores and the one with highest score will be chosen as the best translation. If none of the candidates appears in top 5,000 terms, then no disambiguation is performed and every word strategy will be used.

Failure to translate terminologies The second source of translation error is failure to translate terminologies. Dictionary is often considered as a "shallow but broad" translation source, therefore terminologies for a specific field may be not covered very well in a general dictionary. This kind of error referred to as "out-of-vocabulary" (OOV) error.

OOV terms can be divided into nine categories [16]: named entities, general vocabulary (words that are expected to be found in a comprehensive monolingual dictionary), newly formed words, alternative spellings, domain-specific terminology, abbreviations, loan words (words absorbed from another language with minor spelling changes), transcribed sounds and the last one - an undecidable category for any terms that can't be assigned in any category with certainty.

Among all those nine categories, named entities and domain-specific terminology are considered the most important. Name entities play an important role in IR task for that they take a large percentage of daily queries. Another factor that makes named entity translation an crucial topic is that named entities take up the largest share among all missing terms. Distribution of the out-of-vocabulary words in the CLEF 2000 collection was given in Demner-Fushman [16] and the distribution shows that named entities take up about 50% of missing terms. To handle name entity translation, the most commonly used approach for languages sharing the same writing system is to keep terms untranslated. For those don't share the same writing system, a phonetic translation has been proven to be helpful.

Domain-specific terminology is also considered an important category. Those terms have decisive affects in IR system for that they are highly specific for a search task. The second reason is that unlike other categories whose coverage increases as the size of dictionary increased, coverage for domain-specific terminologies barely change as dictionary size increases.To improve the coverage of domain-specific terminologies, domain specific dictionaries are required.

Simple lexical processing like stemming can be used to reduce OOV error. Bilingual dictionaries usually contain several morphological variants for a term. However there may be the possibility that the variant we are looking for is absent. In this case, applying the same stemming rule on both dictionary terms and query terms then try to match them again will allow certain absent variants to be matched. Oard [11] explored the effectiveness of stemming approach. Stemming is proceeded in the following steps: The first step is trying to find exact match for query term in dictionary. If exact match is not find, every word in dictionary as well as query term will be stemmed then try to match again. If still fails, query term will be remained unchanged. However, the experiment result shows that such method doesn't work well: average precision drops after stemming is applied. Precision drops because of the mismatch caused by stemming. In some scenarios, different variants represent different information needs. Therefore, stemming should be used extra carefully as an approach to help with OOV error for that inappropriate stemming may introduce ambiguity.

Failure to translate multi-word phrases The third is failure to translate multi-word phrases or translating them poorly. Translation for a multi-word phrase can differ a lot from translation for each individual term inside the phrase.

To handle multi-word phrase properly, the first step is to detect them. In Ballesteros [4] a detection approach based on POS sequence analysis is used. First, queries are tagged by POS tagger. Then sequence of nouns and adjective-noun pairs are considered as phrase. Phrase will be translated with strategy that is different from individual term. Ballesteros [4] proposed an approach adopting co-occurrence statistics for disambiguate phrase translation. This approach is based on the assumption that correct translation terms for phrases will tend to co-occur more in target language corpus while incorrect translation terms will not co-occur. It consists of the following steps: first, a potential translation candidate list is obtained for each query term from a bilingual dictionary. The lists are filtered and only candidates having the same POS tag with query term is left. Then if a phrase is detected, for any two continuous terms in the phrase, all possible pairs {a,b} are generated by picking "a" from the first term's translation list and picking "b" from the second term's translation list. The next step is to infer the best {a,b} pair by using co-occurrence statistics. The quality of {a,b} pair is measured by a metric called "net co-occurrence". Net co-occurrence is calculated with the following formula:

$$em(a,b) = max(\frac{n_{ab} - En(a,b)}{n_a + n_b}, 0) \tag{2.1}$$

n_a and n_b are total number of occurrence for a and b in the corpus while n_{ab} represent the times a and b fall into the same text window with a pre-defined length. $En(a,b)$ is defined by $\frac{n_a n_b}{N}$. N is the total number of text windows in the corpus, which can be calculated with total length of corpus / window size. This metric reward rare co-occurrence by normalizing over the total number of occurrence of a and b.

After all possible pairs (a,b) have been assigned a score, the pair with highest score will be selected as the final translation. A notable advantage of this approach is that it doesn't require comparable or parallel corpora. It's conducted on the monolingual corpus that will be searched on.

Another way to translate phrase is to use direct translation resources like a phrase translation table or phrase dictionary. Compared with statistic based method which can be applied on any multi-word phrase, coverage of phrase dictionary based approach is limited. However, translations that found in a phrase dictionary are more accurate than translations obtained by co-occurrence statistics. For compositional phrases, both phrase dictionary and co-occurrence based method are able to provide correct translation. But for non-compositional phrases, phrase dictionary tend to provide much better translation than co-occurrence method does [4].

Query expansions Besides error type specific approaches, query expansion has been proven to be helpful in improving dictionary based translation. Query expansion refers to the process of expanding original query with popular terms from highly ranked retrieved documents. For example, original query "CMU" gets documents about "CMU application", "CMU CS" and "CMU location". The first two documents are ranked higher therefore used for query expansion. Terms "application" and "CS" are selected, therefore we'll have an expanded query "CMU application CS". Compared with the original query, expanded query is more specific and less ambiguous.

Query expansion can be applied before translation, after translation or both. Previous work has shown that pre-translation expansion enhanced precision for that it provides a stronger based for translation, while post-translation expansion increases recall and reduce ambiguity by de-emphasizing irrelevant translation [17].

Mikolov [6] implemented pre-translation expansion on the parallel corpus. First, query in source language was used to retrieve top 20 documents on source language corpus. Then top 5 source terms will be used to expand the query. After expansion, query is translated and used for retrieval on target language corpus.

Post-translation expansion can be done by pseudo relevance feedback. Pseudo relevance feedback is a commonly used approach to improve retrieval performance. It's done by first retrieve top ranked documents with original query and then use most frequent terms in retrieved top ranked documents to expand the original query.

2.3 Neural ranking model

Sate-of-the-art traditional CLIR system is capable of achieving the same accuracy with monolingual IR systems [5]. However, building a high performance CLIR system takes complex work.

For dictionary based CLIR systems, the first challenge is to find a proper bilingual dictionary in machine-readable format. Well-known, high quality dictionaries like Oxford dictionaries are usually limited by copyright issues. While free dictionaries which can be used online or available for download are usually in low quality, with a low coverage terms or nonstandard translations. Another limitation of dictionary based translation is that dictionary usually have a low coverage in certain type of entities: name entities, newly created terms and phrases. Name entities that refers to real-world objects, such as persons, locations, organization, products .etc [18], such as "George Bush", "Carnegie Mellon University". Those name entities are usually not included in dictionaries. Similarly, newly created terms are also hardly covered in a standard dictionary. The third type, phrases, need to be first identified and then translated with phrase table, which also faces a challenge in coverage. The second problem for a dictionary based CLIR system is ambiguity. To resolve ambiguity, a CLIR system adapts mechanisms like concurrence analysis to choose better translation candidate during translation [4]. Also, query expansion is used to further resolved ambiguity in CLIR tasks [17]. All those procedure takes non-trivial work to build and time to run.

Machine translation based CLIR systems also have several limitations. First, since machine translation is studied as independent field, machine translation systems are used as "black box" in CLIR tasks. In a query translation case, a query in source language is fed to the machine translation system and translated query in target language is given by the translation system. Unlike the dictionary bases translation where we can see the translation candidate list and make different strategies to choose candidates, a machine translation system doesn't provide us much freedom for tuning and adjusting. Another problem in using machine translation system for CLIR is that those systems are computationally expensive. Translation large corpus may take a long time.

Neural ranking, as a newly emerging subfield in IR field, brings opportunities to break the limitations in traditional CLIR systems. Many limitations in traditional CLIR system lies in the underlying ranking models. Traditional ranking models uses exact matches signals as relevance evidence, like query contains term "Apple" is related to document contains term "Apple", while neural ranking is able to use soft-match signals, like term "Apple" is related to term "iPhone". Exact-match can be considered as boolean signal because it's either a "match" or "not match", while soft-match is a real value which means a word pairs can have different relevance levels. With soft-match, we'll be able to express the word pairs' relevance relationship with a level between "match" and "not match". This gives the model a lot more freedom in learning and ranking. In neural ranking model, soft-match is achieved by representing words with word embeddings. In the next section, we'll give an introduction about word embeddings and also discuss how to extend the concept of soft-match to a bilingual scenario by using bilingual word embeddings.

Word embedding

Before word embedding, commonly used text representation technique is "one-hot encoding". In one-hot-encoding representation, each word is represented by a large vector of size of vocabulary and only the bit corresponding to the represented word is one while all the other bits are zeros. Such representation is sparse and makes no assumption about word similarity. Compared with one-hot encoding, word embedding is a dense, continuous representation for words. Each word is associated with a real-valued vector. Distance between words' embeddings reflect can reflect certain similarities. This is feature enables soft-matches between document and query in IR tasks. Traditional IR models only consider exact match. Now with word embeddings it's possible to quantize similarity between words by a real-valued number.

A traditional approach of training word embedding is neural probabilistic language model. The learning objective is to maximize the possibility of the next word after observing a window of previous words. Such probability is calculated by assigning a score to each possible words in the vocabulary and takes the form of soft-max to change score into a possibility value. This requires the calculation of soft-max for the whole vocabulary which can be very expensive for large corpus. A more efficient way to train word embeddings is to maximize the possibility of assigning real word as target word given previous words by negative sampling [6], like the word2vec. Both training approaches can obtain word embeddings that encode contextual information, which means words will have similar embeddings if they are surrounded by similar context.

For CLIR tasks, we want word embeddings to encode translation relationships. To achieve this goal, we can use bilingual word embeddings that preserve similarity across language boundaries and further tailor it for retrieval task by training it with the corpus we want to conduct search task on. Pre-trained bilingual word embeddings based on other corpus can be used in CLIR task to preserve translation relationships. Such bilingual word embeddings can be trained efficiently by extending the model of word2vec. By merging and shuffling a bilingual document pair in a comparable corpora, a single "pseudo-bilingual" document can be created [3]. Then use such pseudo-bilingual document to train the word2vec model and the trained word embeddings will be bilingual. Such bilingual word embeddings can be used directly for query translation. For example, English word "Apple" will be considered similar to Chinese word for apple by word embeddings. However, such method requires large bilingual comparable corpora, which takes time to collect and process. Therefore this approach will not be explored in this work.

Unlike bilingual word embeddings which are rare and difficult to obtain, monolingual embeddings are much more common and widely used. If we can align monolingual embeddings between different language pairs to make them comparable, we'll be able to exploit the high-quality pre-trained monolingual embeddings trained on large scale datasets. The alignment between two languages can be achieved by using a small set of bilingual translation pairs [8]. The idea is based on the fact that different language spaces share similar similarity relationship among words. Monolingual embeddings for different languages are like maps for the same country but with different rotation angles [8]: describing the same entity similarity relationship but each is rotated in different angles. Therefore, by randomly picking a small set of words and aligning them, the whole vector space for each language will be aligned. The aligned monolingual vectors are comparable: English word embedding for "mom" will have high cosine similarity with the corresponding Chinese word embedding. In this work, we'll use aligned pre-trained monolingual embeddings for Chinese and English in experiments to explore the efficiency of pre-trained embeddings in bilingual IR task.

Both of the approaches above generate pre-trained embeddings. By pre-trained embeddings we mean they are trained on corpus other than the corpus we can to search on. Such pre-trained embeddings only preserve translation relationships but no relevance evidence. Previous work shows that word embeddings tailored locally for IR task can greatly improve the retrieval performance. Trainable word embeddings that adjusted based on user click feedback is shown to be an important source of effectiveness [2]. Therefore pre-trained word embeddings may not be the optimal choice if used in CLIR tasks without modification. To use them in CLIR, further adjustment based on local corpus is needed.

A third approach to obtain bilingual word embeddings is to train a neural ranking model on a bilingual training dataset, so that the model can learn bilingual word embeddings that encode both translation relationship and relevance evidence. This can be achieved by initializing word embeddings with pre-trained bilingual embeddings and adjust them based on training data or initializing word embeddings randomly and allow a neural ranking model to learn bilingual word embeddings by itself.

Kernel-based neural ranking model

After having words represented with word embeddings, now it's possible to capture soft-match signals. However, exact matches are strong signals for relevance while soft-match are considered weak signals for relevance. Such weak signal has to be processed extra carefully when considered as relevance evidence. For example, when using word embeddings to derive word pair similarity, "London" and "Milan" may be considered similar because they appear in similar context and therefore tend to have similar word embeddings. However document with "Milan" won't satisfy user's information need about "London". Therefore the key lies in how to combine weak signals to produce accurate evidence for relevance.

Many current neural ranking models are designed to solve the problem of soft-match and we use the kernel-based neural ranking model (K-NRM) [2], which is the recent state-of-the-art. In the K-NRM, a set of kernels is used to capture different types of word pair similarities. For example, the strongest similarity level is considered as exact match and there can be other relevance levels for soft-matches. The result produced by kernel layer will be fed into a ranking layer as features for generating ranking scores. K-NRM model is proven to beat the feature-based ranking model and state-of-the-art neural ranking model by 65% as shown in experiment across difference scenarios. This work will adopt K-NRM as neutral ranking model and explore effectiveness of different CLIR solutions. Detailed information about K-NRM will be introduced in section 4.

3

A neural CLIR approach

A neural approach for CLIR is proposed in this section. Our neural CLIR approach tries to give a CLIR solution based on neural architecture by extending the capture of soft-match signals [2] in monolingual terms to the bilingual scenario. This approach is consist of two components: bilingual training data that providing translation knowledge and the underlying neural ranking model, the K-NRM, that learns bilingual soft-match signals from training data.

This neural CLIR approach addresses two largest limitations in traditional CLIR: need careful tuning for high performance and affected seriously by training data. First, by solving CLIR problem in a neural architecture, performance can be optimized via learning. Components that require manual tuning are much less compared with building a high-performance traditional CLIR system. Second, neural CLIR approach is more robust to translation errors compared with traditional CLIR approaches. The most commonly used translation strategy in traditional CLIR is query translation, which means that translation result will be used as queries directly. In this framework, a low quality translation component will lead to low quality translation results, and low quality translation results will result in bad queries. If initial queries are in poor quality, so will be the initial ranking. When initial ranking is in low quality, traditional approaches to improve initial ranking like query expansion won't help since the source of expansion terms are too noisy. On the other hand, instead of using in queries, neural CLIR approach uses translation results in training data. In this way, model can learns how to use translation knowledge by capturing the translation relationships as soft-match signals. This freedom in using translation knowledge gives the model the ability to yield better performance compared with a traditional CLIR system using the same translation knowledge source.

The remaining part of this chapter is organized in the following way: first bilingual training data used in neural CLIR is introduced, then a brief introduction is given to K-NRM, the underlying neural ranking model for this approach.

3.1 Bilingual training data

Bilingual training data can be as a source of translation knowledge by neural ranking model and allows the trained model to take target language query directly as input, which is different from query based translation approach where only target langue can be used as input for ranking module. An example of bilingual training data is shown in Figure 3.1. In Figure 3.1, to generate bilingual training data, queries in monolingual source language training data are translated into target language with machine translation system or other translation approach, while documents remain unchanged.

Bilingual training data approach has several advantages compared with traditional translation approaches.

The first is that bilingual data approach can tailor translation knowledge specifically for IR task while traditional approaches only use translation knowledge but make no modification to it. For traditional retrieval model, translation knowledge from external source (e.g. dictionary, embedding) is used directly for query transla-

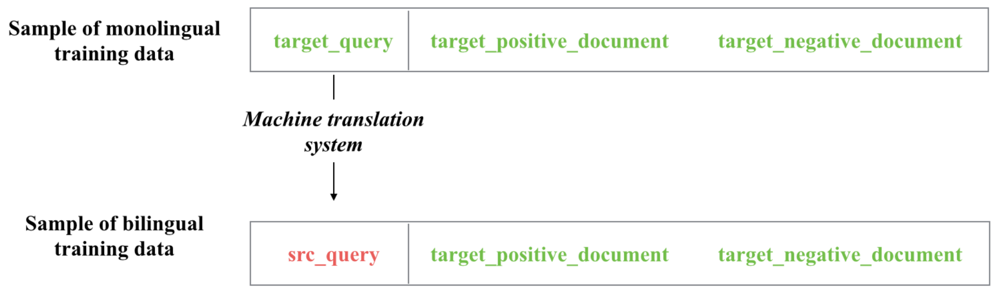


Figure 3.1: Sample of monolingual and bilingual training data.

tion and feedback from retrieval result won't be reflected on translation knowledge. On the other hand, for bilingual data approach, translation knowledge can be encoded implicitly in training data. Model trained with such data will be able to learn both translation and relevance relationship between term pairs. For example, traditional approach knows that "Apple" means a fruit in Chinese based on given translation knowledge. For bilingual data approach, if we provide the model a data entry indicating that English word "Apple" is related to a Chinese document talks about product of Apple the company, the model will able to learn that "Apple" is related to not only "苹果" (apple) but also words like "手机" (phone), "电脑" (computer), "公司" (company), etc. Those smilier relationships learned by the model contains not only translation knowledge but also relevance evidence.

The second advantage is that neural ranking is more robust with low quality translation knowledge. Traditional approach directly uses translation knowledge for query translation. Therefore the initial quality of translated query largely decides the accuracy of ranking. Even though the initial ranking can be improved by query expansion, the extend of such improvement is very limited if a poorly translated initial query is given. For bilingual training data approach, translation results are used as training data. Therefore the translation failure of single item may not affect the final learning result as long as the translation errors are independent and not frequent.

3.2 Neural ranking model

Neural ranking model used in this work is a newly published kernel based neural ranking model (K-NRM) [2]. K-NRM is an interaction based neural ranking model. In K-NRM, words are represented with word embeddings initialized randomly and for each query and document pair, a translation matrix representing word-level similarity is calculated. Then a set of kernels with different weights are used to extract different similarity features from the translation matrix and those features are combined by a learning-to-rank layer to produce the final score. During training, word embeddings and kernel weights will be modified to minimize the pari-wise training loss.

K-NRM is consist of four layers: embedding layer, translation layer, kernel pooling layer and a learning to rank layer.

For a query with n terms, q = $q_1, q_2, \ldots q_n$, and a document with m terms, d = $d_1, d_2 \ldots .d_m$, the **embedding layer** maps the words into L-dimension word embeddings $\vec{v}$. Now query q and document d is represented as the following:

$$q = \begin{bmatrix} \overrightarrow{v_{q_1}} \\ \ldots \\ \overrightarrow{v_{q_n}} \end{bmatrix}_{n \times L}, d = \begin{bmatrix} \overrightarrow{v_{d_1}} \\ \ldots \\ \overrightarrow{v_{d_m}} \end{bmatrix}_{m \times L} \tag{3.1}$$

Then **translation layer** will calculate a translation matrix where each element is the cosine similarity of a query word embedding and document word embedding pair. Translation matrix is in dimension $n \times m$ and each element is defined in the following way:

$$M_{ij} = cos(v_{q_i}, v_{d_j}) \tag{3.2}$$

The translation matrix is fed into the **kernel-pooling layer**. A set of kernels extract features that can be used for ranking from the translation matrix.

The kernels used in kernel-pooling layer is RBF kernel:

$$K_k(M_i) = \sum_j exp(-\frac{(M_{ij} - \mu_k)^2}{2\sigma_k^2}) \tag{3.3}$$

For each kernel, word embedding similarity values close to its mean μ_k will get a higher score. Therefore, a set of kernel with different μ_k will be able to capture different types of soft-match signals. For example, if $\mu = 1$ and $\sigma \rightarrow 0$, the kernel will be equivalent to exact match. If $\mu = 0.5$, the kernel will calculate a score based on how many document, query term pair has a word embedding similarity close to 0.5.

The output of kernel-pooling layer is a set of ranking features, representing how well the query and document matches in terms of different types of soft-matches. Those ranking features are fed into a learning to rank layer and a final score is calculated with them.

A complete K-NRM model can be described as:

$$\begin{aligned}
f(q,d) &= tanh(w^T \phi(M) + b) && \text{Learning to Rank} \\
\phi(M) &= \sum_{i=1}^{n} log\vec{K}(M_i) && \text{Soft-TF Features} \\
\vec{K}(M_i) &= \{K_1(M_i), ..., K_k(M_i)\} && \text{Kernel Pooling} \\
K_k(M_i) &= \sum_j exp(-\frac{(M_{ij} - \mu_k)^2}{2\sigma_k^2}) && \text{RBF Kernel} \\
M_{ij} &= cos(\vec{v_{t_i^q}}, \vec{v_{t_j^d}}) && \text{Translation Matrix} \\
t &\Rightarrow \vec{v_t} && \text{Word Embedding}
\end{aligned} \tag{3.4}$$

Note that every component of K-NRM is differentiable, which allows the model to be trained end-to-end. Both kernel weights and word embeddings will be updated during training. Kernel's weight decides its responsibility. A kernel with positive weight is responsible of collection positive relevance evidence indicating a higher ranking, like "Apple" and "iPhone". While a kernel with negative weight is collecting negative relevance evidence indicating a lower ranking, like "Apple" and "IBM". The larger absolute value a kernel's weight is, the more important it is during ranking because it contain word pairs affect ranking scores the most. Kernels can be interpreted at bins for that each kernel is responsible of collecting soft-match signals from term pairs falling into its range (bin). Word embeddings will be tailored during training and lead to migration of word pairs may from one bin to another.

The concept of word pair similarities can be extended to a CLIR scenario easily for that query, document word pairs q_i and d_j in K-NRM don't have to be in same language as long as their word embeddings are in the same vector space [8]. If proper training data is given, the model will be able to learn word embeddings for terms in different languages that are in the same vector space and comparable to each other. We call such word embeddings "bilingual word embedding" when there are two languages involved. Similarity of bilingual word embeddings can be used as soft-match signals just like monolingual word embeddings because they are comparable. This is what we want to achieve in our neural CLIR approach, to allow K-NRM learn bilingual word embeddings by training the model with bilingual training data.

4

Experimental methodology

This chapter gives an introduction for dataset, ranking models and translation approaches used in experiments of this work.

Two datasets are used in this work. One is search log dataset (Sogou-Log dataset) containing initial ranking for each query and the other one is a larger, cleaner dataset (SogouT-16 B) built for research purpose. In experiments, we hope to mimic the situation where user inputs English queries to retrieve Chinese document. Ideally, a CLIR system based on dictionary based query translation approach, which was the state-of-the-art CLIR approach, should be used to generate initial ranking from SogouT-16 B dataset then use neural CLIR to do the reranking. To guarantee the quality of initial ranking, neural CLIR was experimented on the Sogou-Log dataset reranking initial result produced by a monolingual system, which is a flaw in the experimental methodology of this work. Hopefully, in future work high quality initial ranking from a CLIR system can be obtained on SogouT-16 B dataset and this flaw can be fixed.

To compare neural CLIR approach with approaches in previous works, this work tests different combinations of ranking models and translation approaches. Ranking models used for experiments in this work including traditional ranking model (BM25) and neural ranking model (K-NRM). The translation approaches include dictionary based translation, machine translation and bilingual training data approach designed specifically for neural ranking model.

This chapter is organized in the following way: first two datasets are introduced. Then different ranking models are discussed in retrieval methods section. In the last section, different translation methods are covered and possible combinations of retrieval methods with translation methods are also discussed in the last section.

4.1 Dataset

To evaluate the CLIR approaches in different scenarios, two different datasets are used in this work. The first one is Sogou-Log dataset, a search log sampled from sogou.com, the third largest largest commercial search engine in China. It provides a closer simulation to a real web search environment. The second one is SogouT-16 B dataset. It's a large Chinese dataset provided by sogou for academic study purpose. It's larger and more well-organized compared with the search log data. Detailed descriptions for each dataset are in the following sections.

4.1.1 Sogou-Log dataset

A search log sampled from sogou.com with 96,229 distinct queries.In the search log, each query has the following information: documents displayed by sogou.com, user clicks, dwell time and a relevance score given by sogou search engine. Query, document pair and relevance score will be used as training and testing data for the model, while user clicks and dwell time will be used for relevance judgment. Since our model takes pairwise learning to rank loss, each training data sample includes query, positive document and negative document. Statistical data about Sogou-Log dataset is shown in Table 4.1 [2]. The most frequent 1,000 queries are used as test

	Training	**Testing**
Queries	95,229	1,000
Documents per query	12.7	30.5
Search Sessions	31,201,876	4,103,230
Vocabulary Size	165,877	19,079

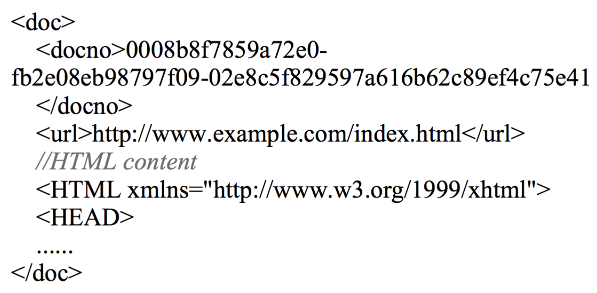
Table 4.1: Statistical data of Sogou-Log dataset.

data and the remaining 95,229 queries are used as training data.

Since this search log is not a well-organized dataset with manually generated relevance judgement, we need to infer testing labels based on click log. Two methods for calculating relevance scores are used in this experiments: Testing-SAME and Testing-DIFF [2]. Testing-SAME calculates relevance scores using DCTR, which is the same click model used for generating training data. While Testing-DIFF uses TACM [9], a state-of-the-art click model for relevance judgement. TACM takes both user clicks and dwell time, and it has been proven to be a good estimation of expert labels on Sogou-Log dataset [9]. In this work, the system is evaluated using both Testing-SAME and Testing-DIFF. Note that Testing-DIFF is considered to be more important because it provides a closer estimation to gold standard labels than Testing-SAME.

4.1.2 SogouT-16 B dataset

SogouT-16 dataset contains 1.17B Web pages sampled from indexed documents of sogou.com. The SogouT-16 requires 81TB before compression. Processing such large corpus requires considerable computation resources. Therefore in this work we'll use the subset of sogouT-16 dataset, which is referred to as "Category B". The category B dataset takes 1.5 TB before compression, comprised by 177,936,163 unique URLs from 818,182 domains.The test collection for this dataset contains 50 manually picked Chinese queries and relevance judgments for those 50 queries. An example for document is shown in Figure 4.1.



```
<doc>
   <docno>0008b8f7859a72e0-
fb2e08eb98797f09-02e8c5f829597a616b62c89ef4c75e41
   </docno>
   <url>http://www.example.com/index.html</url>
   //HTML content
   <HTML xmlns="http://www.w3.org/1999/xhtml">
   <HEAD>
   ......
</doc>
```

Figure 4.1: Sample of SogouT-16 dataset]

4.2 Evaluation metrics

4.2.1 NDCG

NDCG is the short for Normalized Discounted Cumulative Gain. It evaluate the quality of a ranking by awarding highly ranked relevant documents [19]. It's calculated by the following formula:

$$NDCG@k = Z_k \sum_{i=1}^{k} \frac{2^{R_i} - 1}{log(1+i)} \tag{4.1}$$

R_i is the relevance of document at rank i. It can be a multi-valued relevance value, like 0 (non-relevant), 1 (relevant), 2 (every relevant). However, in our Sogou-Log dataset, the relevance judgement is generated with user click, where there are only 0 (not clicked so non-relevant) and 1 (clicked so relevant). Z_k is a factor to normalize the score so that a perfect ranking has $NDCG = 1$ at k [19].

4.2.2 MRR

MRR is the short for Mean Reciprocal Rank. This metric only care about the position of the first relevant document in a ranking. It's calculated by 1 / rank of first relevant document [19]. This metric is helpful especially in web search ranking evaluation because in web search scenario user usually have limited patience when browsing retrieved results and don't care about recall. It's important to have the first relevant document ranked at a high position so that user can find their search target quickly before giving up browsing.

4.3 Retrieval methods

This work will study CLIR on both traditional word-based ranking model and neural ranking model.

4.3.1 Word-based ranking model

Word-based ranking model used in experiment is BM25 model with default parameters. For query with terms $q_1, q_2, \dots q_n$, the BM25 score for a document D is calculated by the following formula:

$$score(D,Q) = \sum_{i=1}^{n} (\log \frac{N - df_{q_i} + 0.5}{df_{q_i} + 0.5}) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_d, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{avgdl})} \tag{4.2}$$

where N is the total number of documents in corpus, df_{q_i} is the number of documents that contain query term q_i in corpus, $f(q_i, D)$ is the term frequency for q_i in document D, $|D|$ is the length of document D in words and $avgdl$ is the average length of all document in the index. k_1 and b are tunable parameters. In the experiment default parameter setting $k_1 = 1.2$, $b = 0.75$.

4.3.2 Neural ranking model

Neural ranking model used in this experiments of work is a kernel based neural ranking model (K-NRM). Details about K-NRM is introduced in Section 3.2.

4.4 Translation Methods

This work mimics a scenario where English speaker uses English query to conduct search on a Chinese corpus and related Chinese documents are returned.

	Traditional retrieval model	Neural ranking model
Dictionary based translation	✓	✓
Machine translation	✓	✓
Bilingual training data	✗	✓

Table 4.2: Retrieval models and translation approaches. The feasibility of different combinations of retrieval models and translation approaches are shown in this table.

Under this scenario, we tested the effectiveness of three translation approaches on two retrieval models. Involved retrieval models has been described in the previous section. Three translation approaches studied in this work are: dictionary based translation, machine translation and bilingual training data.

As shown in Table 4.2, dictionary based query translation can be used in both traditional ranking model and neural ranking model. Dictionary based translation and machine translation are both query translation based approaches. Query translation based approaches requires no change in existing monolingual IR system. Queries are translated into target language by a separate translation module before being fed into the IR system build on target language corpus, which means the IR system is still performing monolingual IR.

Bilingual training data is designed specifically for neural ranking model. It uses training data containing both source language and target language and no explicit mapping relationship from source langue to target langue is needed by the model. In other words, the model doesn't need to know "Apple" and "苹果" is a translation pair and the model will learn translation relationship by itself. But for word-based traditional ranking models, only word level match is considered. As a result, source language term must be mapped explicitly to target language term, like the query based translation approach does. Therefore, bilingual data can't be used by a traditional model as source of translation knowledge.

4.4.1 Dictionary based translation

Dictionary based translation involves in translation source language content in a word-by-word manner. Each source language term is looked up in a bilingual dictionary and a list of translation candidates in target language will be given by the dictionary. For example English term “target” may get a list of Chinese words “目标” (goal), “靶子” (shooting target), ”对准” (aim). Those candidates may have different part-of-speech or with different meanings if the word for translation is polysemous.

Dictionaries used in this work are wiki dictionary and Baidu translation API. The size of Chinese-English wiki dictionary is 7.9G, containing common Chinese and English words. But it lacks information for most of the name entities like organizations and celebrities. To engage the coverage for name entity translation, Baidu translation API is used. Different from Google translation API which provides machine translation functionality, Baidu translation API provides word-by-word translation.

Once obtained the translation candidate list, a target language query can be generated using the list. This can be done by building a nested query with all candidates for a term connected operator “OR” and different terms connected with “AND”.

For example, English query “new target” has two query terms therefore there are two candidate lists. Term “new” has a list of “新的” (latest), “新鲜的” (fresh),”初见的” (not seen before) and term “target” has “目标” (goal), “靶子” (shooting target), ”对准” (aim). Therefore for English query “new target”, the generated Chinese query will be:

(新的OR 新鲜的OR 初见的) AND (目标OR 靶子OR 对准)

4.4.2 Machine translation

Machine translation needs the involvement of machine translation systems. The machine translation system used in this work is Google translation API. Unlike dictionary based translation where a list of translation candidate is given for each source language term, machine translation give a single phrase as result for the whole source language entry.

Two translation directions are used in this work: English to Chinese and Chinese to English. English to Chinese translation is used when machine translation is used as a query translation technique. For example, English query “new target” will get a translation result “新的目标”. Chinese to English translation is used in generating bilingual training data, where Chinese queries in training data are translated into English ones. In this case we will have Chinese query “新的目标” translated into English query “new target”.

4.4.3 Manual translation

Since the goal in testing is to mimic the scenario where English speaker inputs manually written English queries, ideally we should use manually translated English queries in testing. However, due to the large number of testing data, it’s difficult to translate them manually. Therefore, in the experiments we generate the English testing queries using a machine translation system.

There might be a performance gap between real testing scenario and our current experiment setting. This performance gap is caused by the gap between manually translated English queries and machine translated ones. We’ll try to quantify this performance gap in experiment “Testing query translation quality” of Section 6.

5

CLIR on traditional ranking model

In this experiment section, we want to simulate the scenario where user uses English queries to retrieve Chinese documents. English queries are translated from Chinese queries with machine translation or manual translation, depending on the query set size of different dataset. Collection used for retrieval tasks are consist of monolingual Chinese documents.

This group of experiments are conducted on sougouT dataset. Input are 50 English queries manually translated from 50 Chines queries provide by the dataset. The purpose of this experiment group is to build a translingual baseline using query translation on traditional retrieval model. Retrieval model used here is BM25 with default parameter settings.

5.1 Monolingual baseline

In monolingual experiment, original Chinese queries are directly used as input. Since documents in corpus are in HTML format, multiple fields can be extracted when building index. In this experiment we want to investigate effectiveness for different fields and select strongest fields as our foundation for subsequent experiments.

When building the index, we used five fields of documents [21]: body, title, headings, strong and url.

Body field consists of paragraph tags (p tag) and anchor tags (a tag) of HTML file.

Title field is extracted directly from title tag.

Heading field contains h1 to h6 tags.

Strong field is extracted from strong tag.

Url field is extracted from url tag.

In this section, we conducted search on different fields and compare the performance with the NACSIS Test Collections for IR (NTCIR) baseline on the same dataset. The goal for our monolingual experiments is to obtain a similar performance as the NTCIR baseline [21]. In this section, Chinese monolingual experiments are conducted on different fields of the index.

The first experiment uses all fields with the same weight for each field and the final score of a document is calculated by summing up all BM25 scores of each field. The following experiments use one field at a time. For example, the "Body" experiment used only the BM25 scores of the body field, while "Title" experiment used only the BM25 score of the title field.

Results

Experiment results in Table 5.1 show that title field alone yields the best performance and there is a significant gap between the performance of title and other fields. The main problem we found when using other fields is that the recall is very low, which means many relevant documents can't be retrieved. Since our system uses BM25, documents with a high term frequency is preferred. However, in the index there are many similar documents from a same website with a very high term frequency and those document will dominate retrieval result. For example, a query "招聘信

Experiments	**NDCG@10**	**MAP@1000**	**Recall@1000**
NTCIR baseline	0.5961	0.4689	0.91
All fields	0.3358	0.1340	0.45
Body	0.2223	0.0769	0.33
Title	0.5851	0.4364	0.85
Heading	0.2257	0.0463	0.24
Strong	0.1622	0.0207	0.07
URL	0.0163	0.0012	0.01

Table 5.1: Results of monolingual baseline for traditional ranking model. Different fields are used in each experiment. (dataset: SogouT-16 B)

息网" (hiring information website) will retrieve documents from the same website for different locations, like "赶集网-河北招聘信息" (Ganji.com - Hebei (City name) hiring information), "赶集网-天津招聘信息" (Ganji.com - Tianjin (City name) hiring information) ... etc. Those document are very similar and with a very high term-frequency. If no explicit diversification mechanism is implemented, retrieval results will be dominated by those similar documents, which will result in a low recall.

Using Title field helps the most in this problem based on our experiment results. Therefore following experiments are conducted only on title fields and the performance of "Title" monolingual experiment is used as monolingual baseline for following bilingual experiments. Another advantage of using title field is that it's highly computational efficient compared with longer body field or using all fields.

5.2 Dictionary based query translation

This group contains bilingual experiments with English queries as input to retrieve Chinese documents. A dictionary based query translation is implemented in the following way.

Dictionary used in this experiment is wiki dictionary [24]. Each query term is looked up in the dictionary and all equivalent translations are concatenated with

OR operator. For an original query “a b”, the translated query is in the format of (a_1 OR a_2 OR a_3 ...) AND (b_1 OR b_2 OR b_3 ...), where a_1 OR a_2 OR a_3 ... are translations terms for original query term a and b_1 OR b_2 OR b_3 ... are translation terms for term b. If no translation result is found in the dictionary, query term will be left unchanged.

The BM25 model used in this experiment is implemented using Lucene, a free and open-source information retrieval software library [1]. In Lucene BM25, OR operator and AND operator are both implemented by summing up sub-scores. The difference is that AND operator will eliminate documents that don’t contain terms connected by it, which means its a Boolean AND. For example, for query “iPhone AND Apple”, the score for document 1 “iPhone on sale - Apple” is calculated by summing up BM25 score of “iPhone” and “Apple”. On the other hand, document 2 “iMAC on sale - Apple” will not be retrieved for that it doesn’t contain term “iPhone”. Situation will be slightly different for OR operator: for query “iPhone OR Apple” will retrieve both document 1 and 2. The score for document 1 is calculated by summing up BM25 score of “iPhone” and “Apple”, which is in the same way AND operator does, while document 2 will get a score equal to the BM25 score of “Apple” for that it’s the only query term is contains.

To explore the effect of query expansion on dictionary based translation, the following experiments are conducted.

Pre-translation query expansion

To implement pre-translation query expansion on our experiment dataset, we use co-occurrence statistics [4] obtained from WordNet, a large lexical database of English [25].

Pre-translation query expansion is done in the following way. First, a set of

[1] Apache lucene - welcome to apache lucene,” https://lucene.apache.org/.

synonym is get for each query term from WordNet based on co-occurrence statistics. Then all synonyms are concatenated with OR operator. Expanded queries are in the format of (a_1 OR a_2 OR a_3 ...) AND (b_1 OR b_2 OR b_3 ...), where a_1 OR a_2 OR a_3 ...are synonyms for query term a and b_1 OR b_2 OR b_3 ... are synonyms for query term b. The expanded query and original query will be translated with dictionary translation process just described. Expanded query and original query will be combined with AND operator and used for retrieval.

Post-translation query expansion

Post-translation query expansion is implemented with Rocchio's pseudo feedback algorithm [26]. The expanded query is generated based on following formula.

$$q_{expanded} = \alpha * q_{original} + \beta * q_{learned} \tag{5.1}$$

where α is weight factor for original query and β is weight for query learned in relevance feedback. $q_{learned}$ is generated in this way: Each term in top ranked document is assigned a score of $tf * idf$ then terms with high scores are selected for learned query with $tf * idf$ score as term weights.

Pre & Post-translation query expansion

Both pre-translation query expansion and post-translation query expansion are used in this experiment. Initial query is first expanded with WordNet terms, and expanded query is used to obtain a set of initial retrieval result. Then the expanded query in first step is further expanded by adding popular terms selected from initial retrieval result. The final query that has been expanded twice will be used to get a final ranking.

Experiments		**NDCG@10**	**MAP@1000**	**Recall@1000**
Monolingual	Baseline (Title)	0.5851	0.4364	0.24
Bilingual with dictionary based translation	No query expansion	0.1115	0.2124	0.26
	Pre-translation expansion	0.0433	0.0886	0.11
	Post-translation expansion	0.0519	0.0693	0.08
	Pre&Post translation expansion	0.0264	0.0308	0.03

Table 5.2: Results of dictionary based translation CLIR approach on traditional ranking model. (dataset: SogouT-16 B)

Results

Results in Table 5.2 shows that our dictionary based translation system doesn't work well here. One of the cause is that the bilingual dictionary we are currently using is low in coverage for name entities and unpopular terms. Another problem is that our current implementation is not capable of translating phrase. Dictionary based phrase translation requires part of speech analysis for phrase identification and special mechanism for phrase translation, like phrase table or terms' co-occurrence analysis [4]. Since the initial ranking produced by our dictionary based translation system is in low accuracy, the query expansion fails to improve the performance either but makes it even worse. Problem also exists in the query expansion phase. During expansion, WordNet might provide terms that are off-topic in our retrieval task and BM25 is too weak to combine them well.

What can be learned from this experiment is that it takes complex work to build a high-performance dictionary based translation system. First of all, it's hard to find a high quality dictionary with a high coverage for name entities. Consider name entities are very popular in web search queries, failing to translate name entities can cause huge damage in the performance. Another concern is mentioned above, the phrase translation. Last but not least, dictionary based translation need to resolve ambiguity in translations. Query expansion is a commonly used approach. However, as shown in the experiment results in Table 5.2, query expansion can't help in improving performance when initial ranking is poor quality.

5.3 Word embedding based query translation

The goal of this experiment is to explore the effectiveness of using word embeddings as a dictionary-based translation approach. This can be achieved by finding the closest or top n close Chinese terms for each English query term. Word embeddings for Chinese and English terms are pre-trained monolingual embeddings aligned with process described in [8].

Unlike dictionary based approach which is word-by-word translation, word embedding based translation is able to pick best translation with whole query context under consideration. Translation term is scored using the following formula:

$$score(q_i, t) = w_{major} * cosSimiarity(q_i, t) + \sum_{j \neq i} w_{context} * cosSimiarity(q_j, t) \quad (5.2)$$

w_{major} is the weight of the current query term $q_i$for translation and $w_{context}$ is the weight of query terms other than q_i. w_{major} is tuned to be sufficiently larger than $w_{context}$ to avoid twisted translation.

After calculating scores for each query term, candidate translations are ranked and selected based on its score. In this experiment we experimented with top 1,3,5 translation candidates used in translated query respectively.

Results

By checking the generated Chinese queries, we found that considering the context when calculating similarity score does help in finding better translation candidates.

Experiments		**NDCG@10**	**MAP@1000**	**Recall@1000**
Monolingual	Baseline (Title)	0.5851	0.4364	0.24
Bilingual with embedding based translation	Top 1 translation candidate used	0.0371	0.0683	0.08
	Top 3 translation candidate used	0.0459	0.0756	0.09
	Top 5 translation candidate used	0.0463	0.0635	0.08

Table 5.3: Results of embedding based translation CLIR approach on traditional ranking model. (dataset: SogouT-16 B)

For example, when the context is not considered, for query "virtual machine", English term "machine" gets "机关枪" (machine gun) as its top 1 translation candidate, while the whole query "virtual machine" is considered, the top1 candidate is "计算机" (computer). However, from the results shown in Table 5.3 we can see that that embedding based translation doesn't work well either in CLIR. The main problem is that trained word embeddings tend to have high similarity among terms appearing in the similar context but there is no guarantee that the are close in meaning. For example, the Chinese term closed to English term "luna" is "冥王星" (Pluto) for that they are both astronomical objects, while the "月亮" (moon) is ranked in the back and the accurate translation "月亮的" (luna) doesn't even show up in top 10 candidates. Another problem is that bilingual embedding used is trained on another corpus, this may also lead to the situation that the translation candidates picked are not suitable for our dataset. However, most of the monolingual corpus doesn't have parallel corpus in another language to train bilingual embeddings. The approach used in this experiment is to align two monolingual embedding in different languages so that they are comparable [8]. The quality of aligned embedding is not very satisfying considering our experiment result. If more efficient ways for bilingual embedding training can be applied on given dataset, the produced localized word embeddings might be more suitable for IR task.

6

CLIR on neural ranking model

The ability of capturing and using soft-mach signals enables neural ranking model to provide CLIR solution other than query translation. To achieve this, we make the model learn soft-mach signals between terms in different languages from bilingual training data, which we call “bilingual training data approach”. In this section, we first compare our bilingual training data approach with traditional query translation based approaches by testing their performance on the same neural ranking model. Then we conducted a series of experiments to explore the source of effectiveness and error of our bilingual training data approach, which means that the initial ranking is performed by Sogou search engine using a monolingual query in target language.

The following part of this section is organized in the following way. The first part will be focus on evaluating the performance of different CLIR approaches on neural ranking model. A monolingual experiment is run on the neural ranking model as a baseline. Performance gap between the monolingual baseline can show how much accuracy is losing when handling the same job in bilingual environment. Then performance of both traditional query based translation approaches (machine translation and dictionary translation) and our bilingual training data approach on the same

Scenario	**Model type**	**Experiment**		**Train**		**Test**	
				qry	**doc**	**qry**	**doc**
Use ch qry to retrieve ch doc (Monolingual experiment)	Trained monolingual	Monolingual baseline		ch	ch	ch	ch
Use en qry to retrieve ch doc (Bilingual experiment)		Query translation approaches	Dictionary translation	ch	ch	en->ch	ch
			Machine translation	ch	ch	en->ch	ch
	Trained bilingual	Bilingual training data approach (Bilingual baseline)		en	ch	en	ch

Table 6.1: Experiments in performance evaluation section.

neural ranking model, the K-NRM, will be evaluated and compared. The second part of this section will be focus on exploring the possible source of effectiveness and error in bilingual training data approach.

Experiment in this section are conducted on Sogou-Log dataset. To train neural ranking model, an initial ranking should be used as training data and then the trained neural ranking model will be used to rerank the initial ranking. Ideally, we should use the ranking result produced by our traditional CLIR system on SogouT-16 B dataset, which has been described in Section 5. However, due to the fact that performance of our traditional CLIR system is not good enough to provide a sufficiently good initial ranking, we decide perform the training and reranking on Sogou-Log dataset.

6.1 Performance evaluation

Experiments designed for performance evaluation are described in Table 6.1.

There are two scenarios we want to test in this section: user uses Chinese query to retrieve Chinese documents (monolingual experiment) and user uses English query to retrieve Chinese documents (bilingual experiment).

Among the three bilingual experiments, the “Dictionary translation” experiment and “Machine translation” experiment are query translation based approaches. In those two experiments, English testing queries are translated into Chinese with different translation approaches. The last experiment is our bilingual training data approach. Note that previous three experiments, “Monolingual baseline” experiment, “Dictionary translation” experiment and “Machine translation” experiment are all training with monolingual training data, which means the document and query in training data are both in Chinese. On the other hand, the “bilingual training data approach” experiment used bilingual training data where queries are in English and documents are in Chinese. This allows the trained model to handle bilingual input with English queries and Chinese queries directly without query translation.

Since subsequent experiments will be focus on studying and improving the bilingual training data approach, “bilingual training data approach” experiment result here will be used as a bilingual baseline in subsequent experiments.

Those four experiments will be introduced in more detail respectively in the following sections.

6.1.1 Monolingual baseline

The goal of CLIR is to obtain result as good as that of a monolingual IR system. Thus, the first step is to obtain a monolingual baseline serving as a measurement for CLIR. By comparing the accuracy of monolingual result and bilingual result on the same dataset, we will be able to measure how much accuracy is lost in bilingual IR process.

The training data used in monolingual baseline are generated from sogou search log data described in dataset section. Queries and documents are all in Chinese in both training and testing data. 1,000 head queries are used in testing data and 91,313 queries are used in training. All tail queries with frequency <50 have already been removed from training data. Corrupted queries that can’t be decoded are also removed. The training data is shuffled so that the same training queries are not next to each others. Shuffling training data has been proven to improve the model accuracy [2].

6.1.2 Query translation based approaches

To figure out how well traditional CLIR approaches work on neural ranking model, two most common approaches are experimented. By comparing this group of experiments with the following bilingual training data experiments, we will be able to verify whether tailored translation knowledge gained from bilingual training data outperforms the static translation knowledge used by traditional approaches in IR tasks.

This set of experiments are trying to simulate the scenarios where user typing in English queries and hope Chinese documents to be retrieved. Ideally, to mimic actual user behavior, the test queries should be generated manually. But consider the large amount of work, the test queries are translated with machine translation and manually corrected part of the bad translations. More analysis about the quality of those translations can be found in test queries translation section.

In this scenarios, user type into English queries and those English queries are translated by a translation module. Therefore what the neural ranking model sees is still Chinese queries and the model doesn't have to be capable of processing any English data, neither in training phase nor testing phase.

During the training, the model is trained with the same monolingual Chinese training data used in monolingual baseline. For testing, 1,000 English queries are translated into Chinese with a translation module. Two different translation approach are experimented: dictionary-based translation and machine translation.

Dictionary-based translation The English query is translated into Chinese query with word-by-word dictionary based translation. Baidu translation API that provides word-by-word translation service, is used for translation [27]. The reason is that for such web search log dataset, a considerable number of queries involve in name entities which is covered better in online dictionary resources. Baidu translation API returns the most popular used translation for a term. Previous work shows that in dictionary based translation, using a randomly chosen single candidate can be almost as good as using all translation candidates [11]. Therefore we consider using the most popular term will provide a performance close enough to using every candidate.

Machine translation Google translation API, a text sequence based machine translation system, is used for query translation in this experiment [28]. The webpage access of Google translation API has a limitation of 5,000 characters. When the amount of data needed to be translated is large, the API is called in a program so that query number is no longer a limitation.

6.1.3 Bilingual baseline

This experiment used bilingual data for training. The bilingual training data is generated by machine translation without other processing or filtering. This experiment will be a baseline for following experiments that are aiming at analyzing the source of effectiveness and error for bilingual training data approach. Also, it can be compared with traditional translation approach to show which approach is a better fit in neural ranking environment.

In a standard monolingual setting, training data contains Chinese queries and Chinese document pairs while bilingual training data used in this experiment contains English queries and Chinese document document. English queries in bilingual training data is generated by calling Google translation API on all Chinese queries in our monolingual training data.

Since now we have English queries in our training data, the original vocabulary is enlarged. A fact worth noticing is that there is an overlap between Chinese vocabulary and English vocabulary. This is because there are English terms in original Chinese monolingual training data, like abbreviations (e.g. "WOW"), name entities (e.g. "Bing"), commonly used terms (e.g. "game", "make") and some URLs (e.g. "www.sogou.com"). Another source of vocabulary overlap is digits and punctuation marks. In the setting of this experiment, separate vocabulary is used for English and Chinese terms. This design decision is based on previous study that using separate vocabulary for query and document won't affect the performance of K-NRM. With this setting, the model will consider term, namely, "game" in Chinese document different from term "game" in English query. The main problem caused by separate vocabulary is the lack of exact match for those overlap terms. The overlap terms are those terms appear in both query and document. They are supposed to be considered as exact match by the model but now they are considered as different terms. This separate vocabulary problem won't affect the performance seriously because experiments shows that exact match doesn't play an very important role with the current experiment methodology due to the high quality of initial ranking. Exact match related experiments will be discussed in subsequent sections.

As for the testing data, the same 1,000 English head queries translated with Google translation API used in traditional query-translation experiment are also used in this experiment. But it's different from traditional query-translation experiment for that those English queries are directly tested on the model while in traditional query-translation experiment the model still sees Chinese queries.

Experiment	**Testing SAME**			**Testing DIFF**			**Testing RAW**
NDCG	**@1**	**@5**	**@10**	**@1**	**@5**	**@10**	**MRR**
Monolingual baseline	0.2706	0.3719	0.4310	0.3128	0.3624	0.4338	0.3500
Dictionary translation	0.2163	0.3078	0.3802	0.2315	0.2914	0.3751	0.2776
Machine translation	0.2405	0.3305	0.3956	0.2647	0.3212	0.4036	0.3089
Bilingual baseline	0.2996	0.3813	0.4388	0.3382	0.3716	0.4446	0.3623

Table 6.2: Results of different CLIR approaches on neural model. CLIR approaches include traditional CLIR approaches and bilingual training data approach (bilingual baseline). (dataset: Sogou-Log)

6.1.4 Results

Query based translation approaches:

From Table 6.2 it can be seen that in this experiment machine translation outperforms dictionary based approach by 10%. The reason that machine translation based query translation performs better may lie in the vocabulary. During experiments it's found that when dictionary based translation is used, 16% of words in 1,000 translated testing queries are out-of-vocabulary, while this proportion is only 2% for machine translation based approach. This can be explained by the fact that machine translation method is better at taking advantage of context and providing more related translation terms while dictionary based translation provides the most common translation which may not be a good fit in specific context. For example, one English testing query "51 self-study net" is translated into "五十一(fifty one) 自学(self-study) 网络(network)" by the dictionary based translation approach. In this translated query, "网络(network)" is not a proper translation here because query "五十一(fifty one) 自学(self-study) 网络(network)" will retrieve websites introducing network knowledge like protocols, network architecture and so on, while the actual search target is a homepage for a self-study network. However, dictionary based translation can't tell which candidate, "网络(network)" or "网站(website)", is a better choice here. On the other hand, machine translation is able to obtain Chinese translation "51 自学网(website)", which is the perfect translation for this query.

A more general reason for the performance gap between machine translation system and dictionary based translation system may be that today's machine translation systems are trained on large amount of data and also have better coverage on phrases and name entities, which dictionary based system doesn't have. Dictionary based translation system used here is simple without ambiguity resolving techniques while the machine translation system here is highly tuned to produce translation result it has highest confidence on. Actually, it's surprising that dictionary based translation approach can work this well given how simple it is. A possible reason maybe we are doing reranking in this experiment, which limits the lower boundary of how bad the performance can be.

Another fact that worths noticing is that for traditional CLIR approaches, the performance is highly relied on the external translation resources compared with bilingual training approach. This is shown by the fact that in "machine translation" experiment, the queries are translated using Google translation API while the same API is used to generated bilingual training data in "bilingual training data approach" experiment, but bilingual training data approach is able to get higher performance. The reason is that for query translation based traditional CLIR approaches, translation result is directly used as queries. As a result, translation error will directly affect the quality of input query. On the other hand, in bilingual training approach translation results are used in training data, then the neural ranking model learns how to use those translated queries by capturing soft-match signals between an English query term and Chinese document term. Using translated queries in training data is a much more careful way of using translation results compared with using them directly as queries in retrieval.

From the discussion above, we can find two general drawbacks for traditional translation approach. The first one is that the final ranking accuracy is highly relied on the translation accuracy because translated results are used as queries in retrieval process while bilingual training data approach is more robust to translation errors for that it's capable of using translation results in training data and how to use them is learned by the model. The second drawback is that CLIR system using traditional translation approaches usually need extra tuning mechanisms, like query expansion and linguistic processing to achieve high accuracy. Bilingual training data approach can solve those two problem for that can tolerate low quality translation recourses and also doesn't need further processing to reduce translation error. From the result in Table 6.2 we can see that even though bilingual baseline experiment uses the same machine translation system as machine translation experiment does, it's capable of producing a result 20% better than that of machine translation experiment. Further discussion about the effectiveness of bilingual training approach will be coved in next section.

Bilingual training data approach:

From Table 6.2 we can see that bilingual baseline has better performance than traditional translation approach, which supports the assumption that bilingual training data can provide translation knowledge specificity tailored for IR task.

To figuring out the difference between bilingual training process and monolingual training process, we start analyzing the problem by distinguishing the difference in training and testing data.

The first observation is that compared with monolingual training process, bilingual training process lacks exact match. In monolingual environment, same term may appear in both query and document. However, in bilingual environment, even though an English query term has the same meaning with a Chinese document term, they won't be considered as the same term by the model because they are represented by different term id.

The second difference in that after translating Chinese to English, there are uppercase terms and lowercase terms. The Google translation API treat every Chinese phrase as short sentence, with the first letter in capital. Distinguishing uppercase term and lowercase term might be helpful when uppercase term has special meanings that it doesn't have in lowercase form. But in other case, the meaningless "first word starts with capital letter" might bring misleading information and let the model consider a term's uppercase format and lowercase format different even thought they share the same meaning.

The third difference is that different Chinese queries might have the same English translation. As a result, the model will consider documents retrieved by different Chinese queries to be relevant to the same English query. This might be a new source of ambiguity in training, exacerbate the existing ambiguity problem in Chinese queries.

Except performance, another factor should be considered is the computational cost of translation. Translating 855,312 Chinese queries by calling Google translation API takes over 20 hours. If less training data can be used to train the model, both the translation time and training time will be reduced. Considering together with the duplicate translation problem, it may be possible to reduce the size of training data while improve the performance at the same time by cleaning the dataset with certain rules.

All those issues covered in this discussion will be studied on separately in the following experiments.

6.2 Bin analysis

In our neural CLIR, translation relationship like "Apple" is related to "苹果" (Apple) as well as relevance relationship that "Apple" is related to "手机" (phone) is learned by K-NRM and represented by learned word embeddings. Term pairs with different word embedding similarities will be collected by different bins in a K-NRM model. Bins have different responsibilities decided by its weight. Bins with positive weights will contribute positive score to final ranking score because such bins contain word pairs that are considered as strong relevance evidence. Similarly, bins with negative weights contribute negative score to final score and contain word pairs that are negative relevance evidence. Therefore, by analyzing the word pairs in bins of a trained model we'll be able to get some knowledge about what relevance evidence the model has learned and what type of term pairs is considered important in ranking by the model.

One type of bilingual word pairs we are especially interested in is translation pair, such as ("Apple", "苹果" (Apple)). In a monolingual scenario, such word pairs are exact-matches, which are considered as strong signal for relevance. However, there won't be any exact-match in bilingual testing because query and document are using completely different vocabulary as we discussed in previous section, as shown in Figure 6.1.

The same problem of lacking exact-match also exists in training data. Take the data entry shown in Figure 6.1 for example, in monolingual environment the ch_term1 in query will share the same word embedding as ch_term1 in document. But in bilingual scenario, the embedding for en_term1 and ch_term1 are different. Since exact-matches are important relevance signals, losing exact-matches in bilingual scenario might be a source of error.

To verify the assumption above, experiments are designed to explore the effec-

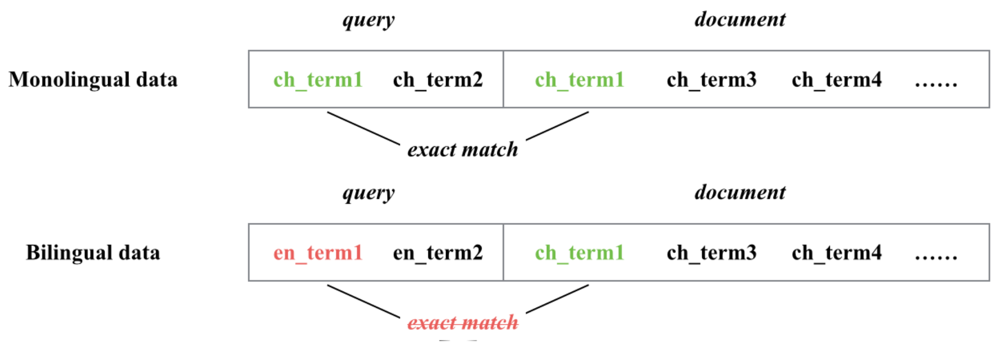


Figure 6.1: Exact-match in monolingual and bilingual data. Shows comparison between monolingual data and bilingual data in terms of exact-match. In bilingual data, there is no exact-mach.

tiveness of exact-match in this training dataset. The first goal is to analyze the contribution of exact match in monolingual environment quantitatively. Such result showing percentage of accuracy decrease indicates the severity of lacking exact match problem. The importance of exact-match in monolingual environment can give us an estimation about how much performance we are potentially losing in a bilingual environment where exact-match is missing.

The second goal of this experiment is to learn how the model treat translation pairs. We'll explore whether translation pairs have similar embeddings after learning as well as how important they are during ranking.

The first goal of studying effectiveness of exact match in monolingual environment can be tested by disabling the exact-match kernel in a standard K-NRM. Recall that in K-NRM, a set of kernels collect different types of soft-matches. The kernel used here are RBF kernels described in Equation 3.3. A kernel with $\mu = 1$ and $\sigma \rightarrow 0$ is equivalent to performing exact-match. By removing that kernel form kernel-pooling layer, we can disable exact-match in the model. Repeat the monolingual experiment with this modified model will yield result that doesn't have exact match effect.

For the second goal, we are interested in how the K-NRM treat translation pairs in a bilingual scenario. To achieve this, we need to observe the word similarity

Experiment	Testing SAME			Testing DIFF			Testing RAW
NDCG	@1	@5	@10	@1	@5	@10	MRR
Monolingual baseline	0.2706	0.3719	0.4310	0.3128	0.3624	0.4338	0.3500
Exact-match disabled	0.2574 (-5%)	0.3620 (-3%)	0.4218 (-2%)	0.3233 (+3%)	0.3696 (+2%)	0.4407 (+2%)	0.3484 (-0.5%)

Table 6.3: Results of exact-match disabling experiment. Shows performance for a monolingual model with exact-match disabled.

pattens of translation pairs assigned to the same bin in a trained model. Since the role of a word pair played in ranking is decided by the bin it belongs to, we first need to assign word pairs to proper bins.

The trained model used for this study is the one trained in bilingual baseline experiment for that it has the best performance so far. Testing data used to generate word pair is also the testing data used in bilingual baseline experiment. For each English query $en\_qry = \{q_1, q_2, ...q_n\}$ and Chinese document $ch\_doc = \{d_1, d_2, ...d_m\}$ in testing data, we generate $n \cdot m$ word pairs (q_i, d_j) for all $i \in (1, n), j \in (1, m)$. Then each word pair is assigned to its closest bin with the way described above.

After assigning testing query and document word pairs to their closest bins, we'll be able to analyze the pattern of each bin. By finding the bin that contain the majority of translation pairs and look at the bin's weight, we'll be able to understand how those translation pairs affect the ranking.

Results

Table 6.3 shows the performance of monolingual training with exact-match disabled. From the results we can see that exact-match doesn't play an important role in this experiment. This may because we are doing reranking of high-quality initial ranking results. Since all entries in the initial ranking contain exact-match signal, it will make exact-match signals less decisive in reranking. After knowing that losing exact-match won't affect ranking accuracy in the current experiment setting, the next problem we want to address is how the model assign bins for translation pairs in bilingual training process.

To further understand how translation knowledge is learned in bilingual training, it's meaningful to know about how word embeddings are learned in a monolingual training process. We conducted the bin-assignment experiment on our monolingual baseline so that we can compare the difference of word embedding learning in bilingual and monolingual environment. Table 6.4 shows the statistical data for 11 bins from the model trained in bilingual and monolingual baseline experiment. By visualizing the word pair counts in Figure 6.2 (a), we can see that the distribution of word pairs across different bins are highly askew. Most of the word pairs are assigned to bin 6 and bin 7 and this is a feature shared by both bilingual and monolingual training process. Since most of word pairs fall into bin 6 and 7, it's not surprising that we find most of the translation pairs in them because most of the word pairs are not important in ranking. Compared with large bins that include the word pairs that are not important, we are more interested in those bins that contain word pairs decisive in ranking. To further understand the effectiveness of each bin, we need to combine the results from the disabling-bin experiment.

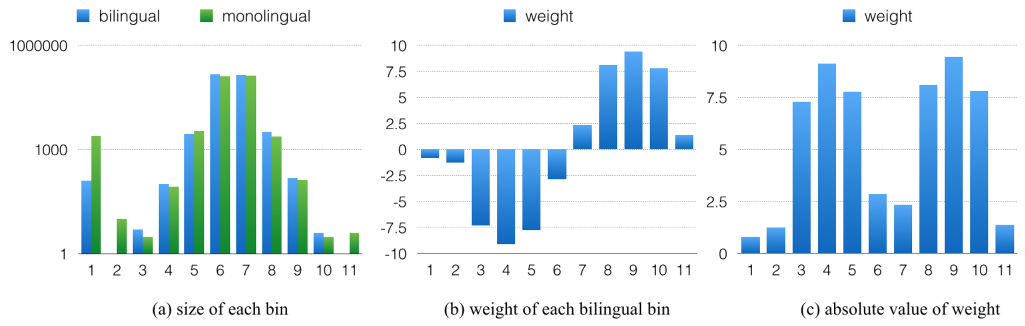


Figure 6.2: Statistical data for bins in K-NRM. Visualization of bins' statistical information in Table 6.4. (a) shows the size of both bilingual and monolingual bins. (b) shows the weight of each bilingual bin. (c) shows the absolute value of each bilingual bin's weight.

Bin		Bilingual		Monolingual	
ID	mean	weight	size	weight	size
1	1	- 0.78	126	- 1.80	2,421
2	0.9	- 1.25	0	- 0.02	10
3	0.7	- 7.28	5	2.54	3
4	0.5	- 9.13	102	18.32	83
5	0.3	- 7.78	2,871	23.16	3,340
6	0.1	- 2.84	146,050	- 2.31	130,670
7	- 0.1	2.34	142,510	- 5.87	134,014
8	- 0.3	8.10	3,232	- 10.00	2,388
9	- 0.5	9.43	154	- 8.78	132
10	- 0.7	7.79	4	- 6.29	3
11	- 0.9	1.37	0	0.11	4

Table 6.4: Distribution of term pairs across kernel bins. Include the mean, weight and size for both bilingual and monolingual systems. Size is defined by the number of word pairs being assigned to a bin.

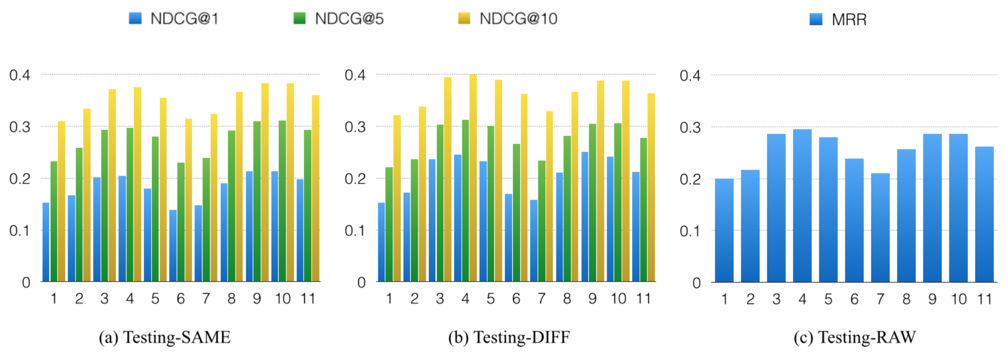


Figure 6.3: Performance of each individual bin in bilingual model. Achieved by enabling one bin at a time and with all the other bins disabled.

Observing performance of individual bin helps in understanding each bin's effectiveness. Figure 6.3 shows the performance of each individual bin with all the other ones disabled. One observation is that the effectiveness of a bin can be reflected by the absolute value of its weight after training. Figure 6.2 (b), (c) visualizes the weight of each bin and it can be seen that the absolute value of a bin's weight and its performance are consistent with each other. This indicates that we can use the weight of a bin to tell it's effectiveness. The larger the absolute value, the more decisive that bin is in ranking. A positive weight means word pairs in this bin implies a higher ranking for that they make positive contribution to the final scores while similarly negative weight implies a lower ranking.

After building connections between bin's effectiveness and its weights, we are able to find most important bins in the model. From Figure 6.2 (b) we can know that the bin with largest positive weight is bin 9 and bin with smallest negative weight is bin 4, which means bin 9 contains word pairs that give a query document pair higher rankings while bin 4 contains word pairs indicating lower rankings. By comparing word pairs in bin 4 and bin 9, a very interesting observation is that there are many similar word pairs in bin4 and 9. Some typical word pair format including (the, a Chinese location name), (weather,a Chinese location name) and (Illegal, a Chinese location name). For example, bin 4 has word pairs (the, 贵阳(Guiyang, a city name)), (the, 兴业(Xingye, a city name)), while bin 9 has word pairs (the, 景德镇(Jingde, a city name)), (the, 黄石(yellow stone)), (the,自贡(Zigong, a city name)). Those similar word pairs are assigned to two opposite bins, which is an important sign showing that word pair similarities are not decided by translation relationship or the effect of translation relationship is negligible. Instead, word pair similarities are decided largely by the preference shown in training data. This explains why documents contain some locations are learned to be ranked higher than the other. Another supportive evidence is that bin 9, which contains word pairs implying high rankings, doesn't contain any translation pair. Actually, most of the translation pairs are assigned to bin 6 and 7. Their weights are much smaller compared with 4 and 9, thus they are considered as noisy bins that can't provide much helpful information in ranking. This further implies that identifying translation pairs won't helping much in deciding final ranking.

All these evidences show that explicitly preserving translation relationship is not the effectiveness source of bilingual training approach. On the contrast, the effectiveness comes from the fact that the constrain of word-by-word translation is broken in the training process. For example, in bilingual training data the model sees an entry with English query containing terms $\{q_1^{en}, q_2^{en}\}$ and Chinese document containing $\{d_1^{ch}, d_2^{ch}, d_3^{ch}\}$. Let's assume (q_1^{en}, d_1^{ch}) is a translation pair, but for the model it's not aware of such constrain and it may learn that (q_1^{en}, d_2^{ch}) has a high similarity according to the training data it sees.

Based on the above discussion we can come to the conclusion that in a reranking task, losing exact-match for translation pairs in bilingual training process won't decrease the performance, on the contrast, without such constrain the model is able to learn word embeddings that are better at distinguishing similar documents and obtain higher ranking accuracy. This can be interpret as a source of effectiveness in neural CLIR. It indicates that the model is capable of learning what is important in ranking, which is a combination of relevance relationships (exact-matches) and popularity relationships. In the current experiment setting, relevance relationships is suppressed by reranking methodology which makes popularity dominate.

6.3 Uppercase lowercase

The Google translation API translated all phrase with the first term uppercase. For example, query "Weather forecast video" has the first term starting with capital letter but it's not a proper noun. This feature makes the English term appears at the the beginning of query different from the same term appears at other locations while they should be considered identical. This uppercase-lowercase problem affects nearly all English queries because they are all translated in this way. Considering the large coverage of this problem, it might affect the performance seriously.

This set of experiments are designed to explore whether "distinguishing uppercase term and lowercase term" brings effectiveness or error. The first goal is to see how will it affect the performance if we eliminate the difference between uppercase and lowercase letters. The second goal is to figure out whether it will be helpful if this "first word starts with capital letter" phenomenon is eliminated.

The first experiment tries to convert all upper case words to lower case. By converting every English word in that dataset to lower case, we can produce a new bilingual data set with all English term being lowercase. The second experiment tries to narrow down the range of "lowercase converting" by only doing converting on the first word of query. In the first experiment every English word is converted into lowercase. Converting all words into lowercase could reduce the vocabulary size by 20%, which may be helpful because we'll have a denser data representation for training data. But the drawback is also obvious: there are many English words that have different meanings when they are in uppercase format compared with its lowercase format. For example, the word "Apple" and "apple" have a slightly different meaning because "Apple" is also refers to Apple the company. In this case, converting "Apple" to lower case will introduce ambiguity in our learning process because now "apple" and "Apple" are not distinguishable. To reduce the potential information lost in lowercase converting, in this experiment we try to set a rule to distinguish words describing name entities that shouldn't be converted into lowercase. The rule is defined as following: the first word of a query is converted to lowercase when and only when the second word is in lowercase. For one-word query, no converting is performed. This rule is based on the observation that name entities in query are usually have multiple words. The only false negative case of this rule is where a single word name entity appears at the beginning of the query, like "Jiangsu traffic learning network" and "Shenzhen weather". By randomly checking 50 training queries, only 2 false negative are detected. This false negative rate maybe not be low enough to be negligible, but considering the simpleness of the rule, this is an relatively effective way to do the converting. Using this rule, we can identify meaningless uppercase words introduced by translation API. For example "The live video" will be converted

Experiments	**Testing SAME**			**Testing DIFF**			**Testing RAW**
NDCG	**@1**	**@5**	**@10**	**@1**	**@5**	**@10**	**MRR**
Bilingual baseline	0.2996	0.3813	0.43881	0.3382	0.3716	0.4446	0.3622
All to lowercase	0.2650	0.3697	0.43437	0.3163	0.3660	0.4353	0.3436
First word to lowercase	0.2772	0.3786	0.4364	0.3070	0.3599	0.4318	0.3471

Table 6.5: Results of lowercase converting experiment.

to "the live video" while no converting will be done on query "Hong Kong Jockey Club information".

Note that in all experiments, training and testing queries are processed in the same way. If testing queries are in lower case, the second experiment can be performed because upper-lower case information is lost in testing data and we can't add this information back without using name entity identification techniques.

Results

Results in Table 6.5 show that none of the lowercase converting attempts improves the performance compared with bilingual baseline. This indicates that "Setting the first English word in query to be uppercase" is actually a source of effectiveness for the bilingual training process. The thoughts behind this is that a word's location information is preserved in its uppercase format (uppercase indicating it's the first word in query) and such location information is helpful in ranking. This can be observed by comparing the bilingual baseline performance with the "conditional first word to lowercase" experiment. In experiments, we assume there are two types of uppercase word: the first type is word in name entities and the second type is word appear as the first word of a query. In bilingual baseline, we have both of those two types of uppercase words, while in "conditional first word to lowercase" experiment, we tries to preserve the first type and eliminate the second type. The decrease in performance after eliminating the second type of uppercase words indicating that the location information "first word in query" may be a helpful factor in the ranking process. A commonly seen case is that knowing a word's location in the query can provide information about whether it's modified by an adjective. For example, word "room" if appear in the middle of a sentence, it's often following term "live" and used as a phrase "live room". When it appear at the beginning of the query as "Room" it's often referring to a more general concept. Even though the improvement brought by such capacity of distinguishing minor term difference is not very large (about 5% in this experiment), it's still a method worth trying considering its simple procedure.

Another interesting finding is that experiment "all to lowercase" outperforms experiment "conditional first word to lowercase" in Testing-DIFF, which indicates that setting all words to lowercase does help the model generalize better. But compared with the information lost in such converting (name entity and location information), the gain is still not large enough to improve the performance over the baseline.

6.4 Cleaning training dataset

The goal of this experiment set is to explore how to clean the dataset by identifying and removing low quality training data. By observation the training dataset, we found two signs for noisy training data.

The first sign is duplicate translation. Duplicate translation refers to the situation where more than one different Chinese queries ($ch\_q1, ch\_q2...$) are translated into the same English query (en_q). This will introduce ambiguity into the training because such training data will tell the model that documents related to $ch\_q1, ch\_q2...$ are now all related to en_q, even though $ch\_q1, ch\_q2...$ have different meanings.

There are several scenarios where this could happen.

The first type is multiple Chinese queries are name for the same entity but written in different ways. Those terms are usually Chinese names therefore have no corresponding English translation. Google translation will translate those names by Pinyin in this case. Since they have the same pronunciation, they will be translated into the same English query. For example, Chinese query "盛朗熙" and "盛郎熙" will both be translated into "Sheng Langxi".

The second type is synonym. When there are several Chinese queries describing the same concept or object, they will be translated into the same English query. Abbreviations also belong to this category, like "北邮" and "北京邮电大学" will both be translated to "Beijing University of Posts and Telecommunications".

The third type is English terms mixed in Chinese queries. For example, in monolingual training dataset we have "LOL", "Lol", "lol". They are all translated into "lol" by the translation API.

Most of those duplicate translations are harmless because those Chinese queries are considered as identical for users. However, it's worth noticing that one important reason that those Chinese queries are translated into the same English translation is that they are very short. By checking the English queries that have multiple corresponding Chinese queries we can see that they are usually short (no longer than 3 English words) and most of them are single word queries, like "上" (on), "在" (on), "如何" (how), "怎么" (how). It can be observed that many of those Chinese queries are very ambiguous and don't make much sense by themselves. The ambiguity lies in Chinese queries are further enlarged after being mapped into the same English query. Those queries usually have extremely long initial retrieve result list, which is the second sign for noisy training data.

Long retrieved document list is also considered as source of noise because such document list tend to cover a very diverse of topics and may provide confusing knowledge to the model. For example, English query "all" has a document list consist of content about "whole-wheat bread", "complete edition" for some popular novels, a video website with slogan "everyone is the director for his own life", topics

Statistics	Value
average	15,499
minimum	3
maximum	579,738
standard deviation	76,158

Table 6.6: Statistical data for ambiguous queries' document list lengths. Ambiguous queries are defined as multiple Chinese queries with same English translation.

about "full-stack engineer". Those problematic English query are usually the terms that only be meaningful when considered with context. From the example we can see that with such mixed document list, the learning can be very confusing for the model. However, those terms are usually very popular terms that appear in a large number of queries. Consider how serious the problem might be and how many queries they are affecting, duplicate translation queries is a problem should be addressed on.

Another reason for long documents list is personalization and localization in Sogou search engine. Personalization and localization will give different user different retrieval result based on their search history or location. For example, if a user search "pizza", he will get webpages about pizza store near by him or based on his preference which can be inferred from his search history. Therefore different users will get a different list of pizza stores. During training data generating, retrieved document list for the same query will be merged into one. If that query has been personalized or localized for different users, a long document list will be produced after merging. This is also a case where we want those data entered to be removed because we want the model to learn more relevance relationship instead of popularity factor decided by personalization and localization.

After knowing about the two signs for noisy training data, we can use them as filtering rules to identify noisy training data entries and remove them to clean our training dataset. In this experiment, we user those two rules to filter the bilingual training data used for baseline experiment.

List length range	**Number of ambiguous queries**
1 - 100	95
100 - 200	65
200 - 1,000	91
above 1,000	65

Table 6.7: Statistical data for ambiguous queries's distribution. Shows different document list ranges and how many ambiguous Chinese queries fall into that range. A Chinese query is considered to be ambiguous if there are multiple Chinese queries translated into the same English query.

First, we identify all Chinese queries with duplicate translations as ambiguous queries that are potential candidates for removal. Then we decide which candidates to remove by setting different threshold on document list length: if a Chinese query's document list exceeds that threshold length, its corresponding English query will be removed from training data.

To find proper document length thresholds, distribution of document list length could be a helpful source of information. Table 6.6 shows statistic information for document list length of all Chinese queries. From Table 6.6 it can be seen that the distribution is highly askew. We set length thresholds at 100, 200, 1,000 so that each document list length range has a similar number of queries, as shown in Table 6.7.

After having the thresholds, we can remove ambiguous queries from training data. The policy we used here is to remove the English query as long as one of its Chinese translation has a document list exceeds length threshold. For example, English query "all" has corresponding Chinese queries "全","都", "全部", "所有", which all mean "all" after translated into English. Chinese query "全" has a document list size of 507, "都" has a list size of 241, "全部" has a size list of 146, "所有" has a list size of "64". If the threshold is set to 200, English query "all" will be considered as ambiguous and removed because it has two Chinese queries with list size larger than 200. The removal condition is set to be "with one or more Chinese queries' list length larger

Threshold	Training data size	Percentage
No entry removed	8,553,124	100.00%
100	7,390,119	86.40%
200	7,408,823	86.62%
1,000	7,446,192	87.05%

Table 6.8: Training data sizes of cleaning training dataset experiments. Size is measured by the number of (query, document) pairs. Percentage compared with whole training dataset where no entry is removed is shown.

than threshold" rather than "all Chinese queries's list length larger than threshold" because as can be seen in the example, as long as one of the Chinese translation is ambiguous, the English queries itself is ambiguous, even though not all Chinese queries got a very large document list size. The fact is, when duplicate translation happens, more ambiguity is introduced into the query. The duplicate English query is more ambiguous than any of its Chinese queries. Therefore, the ambiguity of an English query should be measured with its worst case Chinese query, which has the longest document list.

With the above setting, we have three experiments, where the English queries with any Chinese translation's document list length larger than 100, 200 and 1,000 are removed respectively. The training data size for each experiment is shown in Table 6.8. It can be seen that a considerable number of training data is missing compared with baseline, which will make the results incomparable with baseline performance. To evaluate the actual effect of removing duplicate, a set of comparable experiments is needed.

Another three experiments are conducted as control group. The training data in these experiment is generated by randomly sampling training data samples from bilingual baseline experiment so that they have same data amount with that in experimental group. Therefore there are six experiments in total, including three experimental groups and three control groups.

For the testing part, the same 1,000 testing queries used in bilingual baseline is used. All six experiments are tested with the same testing data so that comparable results can be obtained.

Results

The experiment results are shown in Table 6.4. By comparing the control group where data entries are randomly removed and experimental group where ambiguous queries with long document list are removed, it can be seen that a document length threshold at 100 makes a small improvement in performance while threshold at 200, 1,000 decreases the performance slightly. Another way to look at the results is comparing only among the experimental group. By making comparison among experiment group results, it can be seen that training data size increases while threshold increases, but the performance is decreasing. As threshold increases from 100 to 10,000, more data entries are included but the performance is dropping. This indicates that the newly introduced data entries are in low quality and bring confusing knowledge to the learning process. In other words, training queries with document list length larger than 100 are too ambiguous to provide helpful information during training.

What revealed by results of this experiment is that it's possible to clean training dataset by removing ambiguous queries. Experiment with threshold 100 uses only 86.4% of the whole training dataset but it's able to get nearly 99% of the performance in Testing-DIFF evaluation. This indicates that 1,163,005 (13.6% of whole dataset) data entries removed from the whole dataset are in low quality and didn't provide much helpful knowledge in learning process.

6.5 Training data size

Observation on how ranking performance changes with training data size can reveal a lot of interesting information. If we can observe a increasing trend in performance as

Experiment		**Testing SAME**			**Testing DIFF**			**Testing RAW**
Remove qry with doc list longer than		**NDCG**			**NDCG**			**MRR**
		@1	**@5**	**@10**	**@1**	**@5**	**@10**	
100	Ambiguous qry removed	0.2603	0.3656	0.4193	0.3280	0.3669	0.4389	0.3581
	Randomly removed	0.2276	0.3357	0.4015	0.2716	0.3357	0.4103	0.3112
200	Ambiguous qry removed	0.2472	0.3599	0.4217	0.3096	0.3647	0.4341	0.3427
	Randomly removed	0.2575	0.3608	0.4190	0.3163	0.3571	0.4344	0.3424
1,000	Ambiguous qry removed	0.2558	0.3713	0.4266	0.3134	0.3613	0.4330	0.3424
	Randomly removed	0.2602	0.3650	0.4250	0.3176	0.3632	0.4376	0.3494
No entry removed		0.2996	0.3813	0.4388	0.3382	0.3716	0.4446	0.3623

Table 6.9: Results of cleaning dataset experiment. Shows performance of models trained with ambiguous queries removed at different document list lengths and a control group with same data amount but data entries are removed randomly.

training dataset size increases, we can then infer that more training data is possible to improve the model's performance. Another question we are interested in is the robustness of model. If the performance is stable while training data size changes, we'll be able to use limited amount of training data to train a model with acceptable performance. To explore those questions, we experimented with different training data sizes in this section.

First, to figure out whether there is an increasing trend in performance as training data size increases, we conducted four experiment with training data size 2,000,000, 4,000,000, 6,000,000 and 8,000,000.

The training data is generated by cutting bilingual training data used in baseline experiment at different length. As shown in Figure 6.4, original training data in divided into four partitions of size 2,000,000 and one last partition with size 533,124. Each experiment uses different partitions. Experiment 1 uses containing data entry

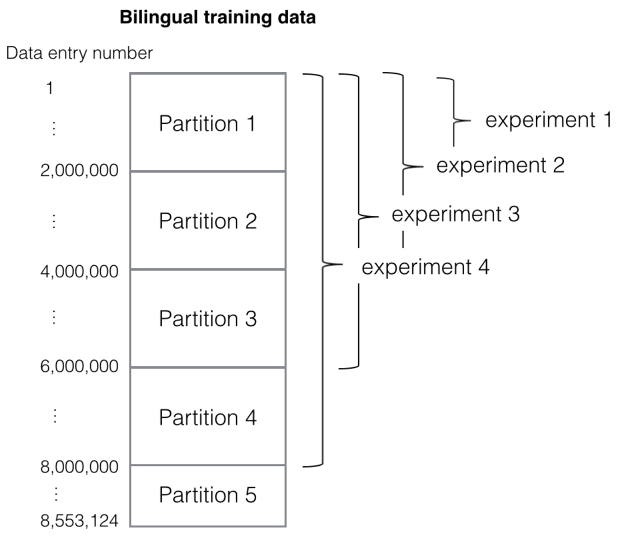


Figure 6.4: Training data generation for training data size experiments.

1 to 2,000,000, experiment 2 uses data entry 1 to 4,000,000, experiment 3 uses data entry 1 to 6,000,000, experiment 4 uses data entry 1 to 8,000,000.

Besides the four experiments above, to further explore the robustness of bilingual training process with small training dataset, we also tested the model performance with training data size 500,000, 1,000,000, 1,500,000 and 2,000,000.

For testing data, the same 1,000 testing queries used in bilingual baseline is also used here so that the result is comparable to the baseline experiment.

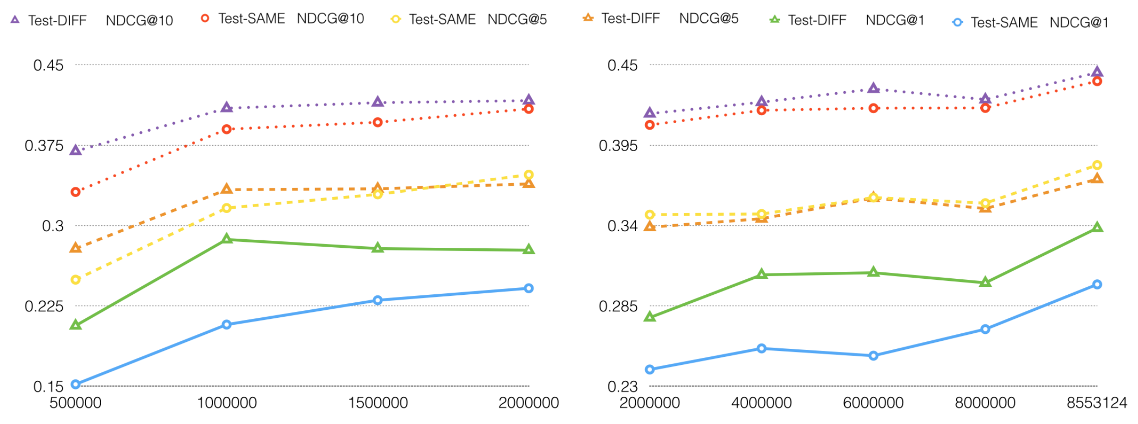
Results

The experiment result for different amounts of training data are shown in Table 6.10. To get a clearer view about the changing trend of performance, we can draw a curve from those data as shown in Figure 6.5.

By observing the curve in Figure 6.5 (b), we can draw to the conclusion that if more bilingual training data is given, the model will tend to yield better result. From Figure 6.5 (b) we can see that there is a general growing trend in performance as the amount of training data increases. To further understand the relationship

Experiments	**Testing-SAME**			**Testing-DIFF**			**Testing-RAW**
	NDCG			**NDCG**			**MRR**
Data size	**@1**	**@5**	**@10**	**@1**	**@5**	**@10**	
500,000	0.1517	0.2494	0.3313	0.2064	0.2783	0.3693	0.2478
1,000,000	0.2075	0.3163	0.3898	0.2869	0.3334	0.4094	0.3106
1,500,000	0.2302	0.3290	0.3963	0.2784	0.3342	0.4146	0.3122
2,000,000	0.2414	0.3474	0.4087	0.2769	0.3388	0.4166	0.3175
4,000,000	0.2559	0.3479	0.4188	0.3062	0.3447	0.4244	0.3347
6,000,000	0.2509	0.3591	0.4203	0.3077	0.3588	0.4334	0.3460
8,000,000	0.2690	0.3553	0.4205	0.3008	0.3516	0.4262	0.3421
8,553,124	0.2997	0.3814	0.4388	0.3382	0.3717	0.4447	0.3623

Table 6.10: Results of training data size experiment.



(a) Performance on smaller data size

(b) Performance on larger data size

Figure 6.5: Results of training data size experiment. Shows performance with different amounts of bilingual training data. X-axis: Number of data entries used in training. Y-axis: NDCG@5-10 for Testing-DIFF, Testing-SAME and Testing-RAW.

between training data amount and performance, we are also interested in the growth of vocabulary size. Figure 6.6 show how the vocabulary size changes with the training data amount. It can be seen that the vocabulary growth follows Heap's law but with a more flat plateau region. This may be caused by a limited topic coverage of the search log data.

By comparing Figure 6.5 (b) and Figure 6.6 we can see that the growing trend of performance is not highly related to the growth of vocabulary size. There is a

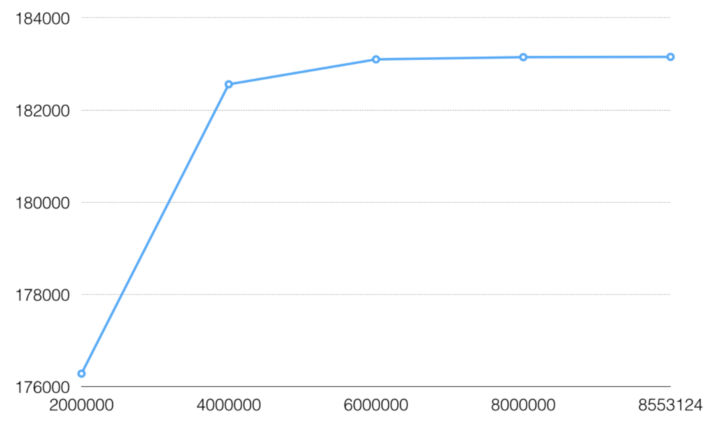


Figure 6.6: Vocabulary size for different training data amount. X-axis: Number of data entries used in training. Y-axis: Vocabulary in number of appeared distinct terms.

sharp increase in performance when the training data size grows from 8,000,000 to 8,553,124 while the vocabulary size only grows by 6 words during this period. To sum up, as training data size increases, the changes in vocabulary size is very small while there can be a burst increase in performance. This indicates that the improvement in performance is not caused by learning new terms but by a finer tuning for word embeddings of learnt terms.

A further conclusion is that larger training data can increase the ranking accuracy and the training dataset doesn't have to cover a wide range of vocabulary. Because it's highly possible that words that play a decisive role in ranking has already been learnt in the beginning of learning.

Another concern other than improving performance by increasing training data is the robustness of the model with small training dataset. By observing Figure 6.5 (a) we can see that the performance at small training data set is not very bad even compared with the best result that can be achieved. From experiment results shown in table 7 it can be calculated that with a training set of 2,000,000, we can get 80%, 91%, 93% of NDCG@1, NDCG@5 and NDCG@10 respectively in both Testing-DIFF and Testing-SAME. This indicates that the bilingual training process is robust even with small amount of training data. The previous discussion that words playing a decisive role in ranking has already been learnt in the beginning of learning, also supports this result. For a training set of size 2,000,000, it's only 20% of the whole training data but 96% of whole vocabulary is already observed. There is a large terms included in a small share of training data are sufficient for producing an acceptably good results.

Another observation on small training data size experiment results shown in Figure 6.5 (a) is that there is a sharp increase from data size 5,00,000 to 1,000,000 and performance stays stable after that. This further supports the conclusion that the training process can be done with a small bilingual dataset. Because the ranking accuracy rises rapidly to a relatively high standard after seeing a small share of training data.

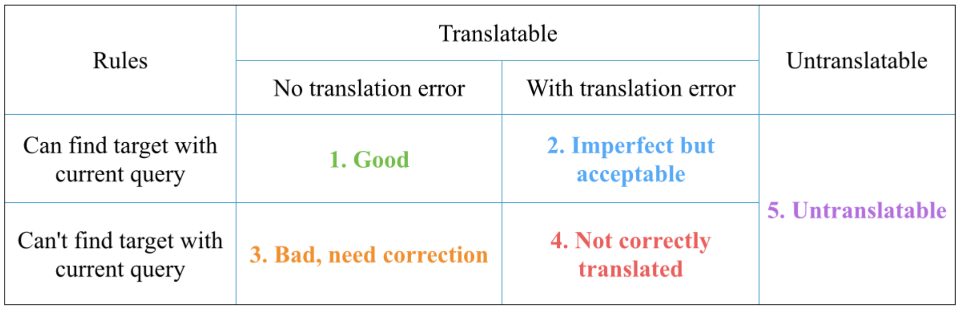
To sum up, increase amount of bilingual training data can increase the performance but the improvement is slow and not linear to the data set size. There are plateaus and small burst of increase could happen when training data size growing. Such burst may be caused by appearance of new terms that are important for ranking as more training data is seen. Besides, the model is robust with a small amount of training data. A subset with size 20% of the whole training dataset can yield 80% to 90% of the performance of a whole training dataset produces.

6.6 Testing query translation quality

Ideally, testing queries should be translated manually to mimic the real scenario where a English native speaker is trying to retrieve Chinese document by inputting English queries. However, due to the large amount of testing query, currently our 1,000 testing Chinese testing queries are generated with the same machine translation system (Google translation API) used in bilingual training data generation.

Since the bilingual training data is translated using the same machine translation system, the vocabulary and expression tend to be more similar in our English training and testing queries compared with real scenario. Clearly, there is a gap between machine translation and real user input, which indicates a potential gap between real scenario performance and current testing performance. To quantify this gap, we conducted a group of experiments to analyze the quality of testing query translation.

To quantify query translation quality, the first step is to define categories for different translation quality levels as well as rules to assign a query to a proper category. As shown in Figure 6.7, we first divide Chinese queries into translatable and untranslatable. Untranslatable here is defined as objects that may not been known by English user and don't have proper English translation, like unpopular websites that don't have official English names, social network usernames and Chinese software, game, book, manga names. The translatable queries are further dived into four categories: "Good", "Imperfect but acceptable", "Bad, need correction" and "Not correctly translated". Two rules are used for deciding a category: whether target content can be found with current translation and whether there is translation error. Next we'll define those two rules more clearly by giving explanation for those two dimensions.



Rules	Translatable		Untranslatable
	No translation error	With translation error	
Can find target with current query	1. Good	2. Imperfect but acceptable	5. Untranslatable
Can't find target with current query	3. Bad, need correction	4. Not correctly translated	

Figure 6.7: Categories of translation quality. Shows five categories for translated query based on translation quality and rules for classification.

During translation, there might be some drifting from the target described in original query and the first rule is aiming at measuring such drifting. In experiments, it's decided by feeding the translated query into a large commercial search engine, Google, to see whether the target described in original query can be found. For example, a French movie with Chinese name "狂爱仪式" is translated into "Mad love ceremony" which is the correct literal translation but the movie can't be find by query "Mad love ceremony" in Google for that its actual name is "Ceremonie d'amour". In this case where a large commercial search engine can't find the target described in original query, we consider this translation as "target content can't be found with current translation". Note that name entities are especially sensitive to such drifting, since a minor difference from the well-known official name can lead to completely different search results.

The goal of the second rule is trying to classify query translations based on the correctness of translation system. Note that after dividing queries based on the first rule, we have two classes now and one class is obviously in better quality than the other. To make sure categories are dividing translation quality evenly, we adopt different definitions for translation error in each class. For the class that can't find target, a looser definition is used where translation error includes failure of name entity translation, missing terms from original query and choosing wrong meaning in polysemous word translation. On the other hand, for the class where target can be found, the translation in this class tend to be in high quality therefore we use a more strict definition for translation error here: besides the translation error just defined, translations that are acceptable but can be improved are considered having minor translation error. For example, query "写信格式" in translated into "Write letter format" which is acceptable but a more natural way to write it is "Sample letters formats".

To define each type of translation more detailedly, failure of name entity translation refers to the situation where a name entity is translated in a word by word manner, for example, “小潮(Xiao Chao)” is a person’s name but it’s translated into “Small tide”, which is a word by word translation for the name. What worth noticing is a special case of name entity translation failure where a reasonable translation is given but it’s different from the entity’s official English name, for example, “终极一班”, which is a series television show, is translated into its literal meaning: “The ultimate class”. However, the show has an official English name “KO One”. Then show can’t be found by the given translation but it’s not considered as a translation error because there is no error in the translation in literal level. This type of drifting error is measured by the first rule as described in the above paragraph. Missing terms from original query refers to the situation where some query terms in original query are not translated and missing in the given translation. For example, query “qq 飞车官方下载(qq racing game official download)” is translated into “qq official download” where the keyword “racing game” is missing. Such error usually happens when the translation API encounters some expression it doesn’t understand. Interestingly, since keywords are missing, those type of translations are mostly can’t be found by commercial search engines. Choosing wrong meaning in polysemous word translation usually result in translations that don’t make sense. One common example is that Chinese term “网” is usually translated into “network” while the correct translation in most of the queries’ context should be “website”.

Figure 6.8 shows a complete view of how translatable queries are assigned to the four categories. With rules and categories defined above, we go through each translated English testing queries in order and assign it to a category based on the rules. This process is repeated on 265 queries. Since no query is skipped in this process, the size of each category can be considered as an estimation for the real translation quality level distribution for the whole testing query set.

Now we have one category of untranslatable queries and four categories of translatable queries. For category “Good” and “Imperfect but acceptable” and “untrans-

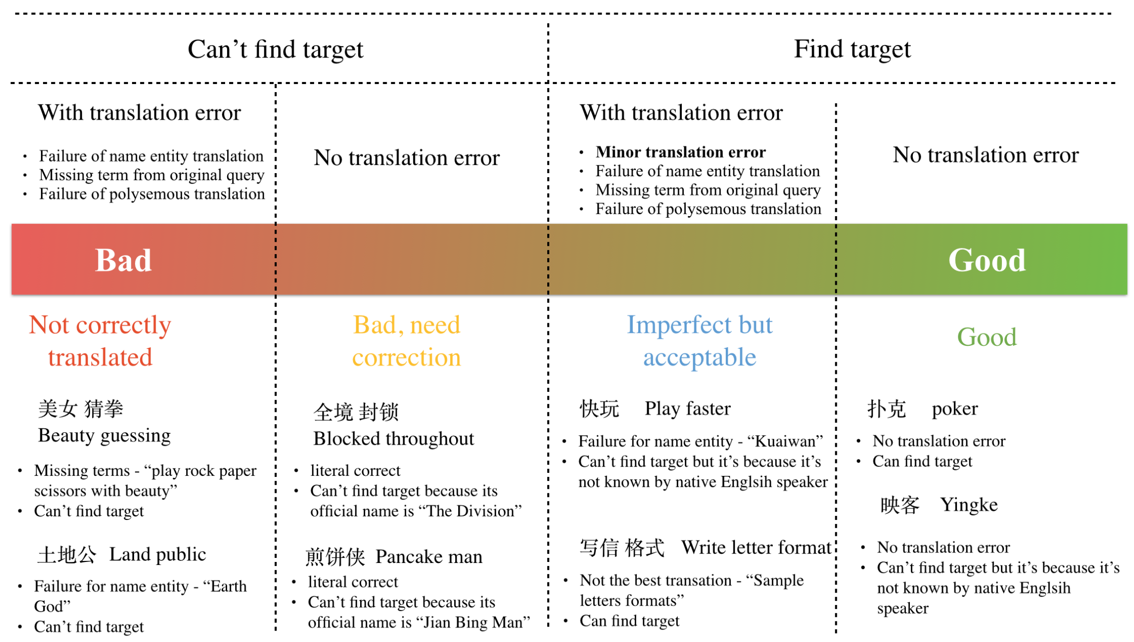

Can’t find target
Find target
With translation error
Failure of name entity translation
Missing term from original query
Failure of polysemous translation
No translation error
With translation error
Minor translation error
Failure of name entity translation
Missing term from original query
Failure of polysemous translation
No translation error
Bad
Good
Not correctly translated
Bad, need correction
Imperfect but acceptable
Good
美女 猜拳
Beauty guessing
Missing terms - “play rock paper scissors with beauty”
Can’t find target
土地公 Land public
Failure for name entity - “Earth God”
Can’t find target
全境 封锁
Blocked throughout
literal correct
Can’t find target because its official name is “The Division”
煎饼侠 Pancake man
literal correct
Can’t find target because its official name is “Jian Bing Man”
快玩 Play faster
Failure for name entity - “Kuaiwan”
Can’t find target but it’s because it’s not known by native Englsih speaker
写信 格式 Write letter format
Not the best transation - “Sample letters formats”
Can find target
扑克 poker
No translation error
Can find target
映客 Yingke
No translation error
Can’t find target but it’s because it’s not known by native Englsih speaker

Figure 6.8: Rules for translatable queries classification. Includes details for assigning categories for translatable queries and example queries for each category.

latable queries”, we consider there are no gaps between those translations and real scenario. The first two category are considered no gap because they are in relatively good quality and the third category “untranslatable queries” is also considered no gap for that in real testing scenario English users are not likely to search for objects in those qureis. Because there is no gap for those three categories, no manual correction is done on them. We only care about bilingual performance and monolingual performance of those queries. It’s done by testing Chinese queries and their English translations in that category on trained models, which are our monolingual baseline and bilingual baseline.

For category “Bad, need correction” and “Not correctly translated”, we assume there is gap between those translations and real queries from an English native speaker. Therefore, we created a list of corrected queries for those two categories individually by manually correcting queries in them. Table 6.11 gives a complete view for experiments in this section.

Category	**Experiment**	**Description**					**Size**	**OOV**
		Train		**Test**				
		doc	**qry**	**doc**	**qry**	**qry manually corrected?**		
Good	monolingual	ch	ch	ch	ch	✗	103	4/221
	bilingual	ch	en	ch	en	✗		4/222
Imperfect but acceptable	monolingual	ch	ch	ch	ch	✗	50	0/155
	bilingual	ch	en	ch	en	✗		4/159
Bad need correction	monolingual	ch	ch	ch	ch	✗	46	0/133
	bilingual	ch	en	ch	en	✗		0/135
	corrected bilingual	ch	en	ch	en	✓		16/131
Not translated correctly	monolingual	ch	ch	ch	ch	✗	38	0/124
	bilingual	ch	en	ch	en	✗		2/132
	corrected bilingual	ch	en	ch	en	✓		21/150
Untranslatable	monolingual	ch	ch	ch	ch	✗	28	0/97
	bilingual	ch	en	ch	en	✗		1/97

Table 6.11: Experiments for translation quality analysis.

Results

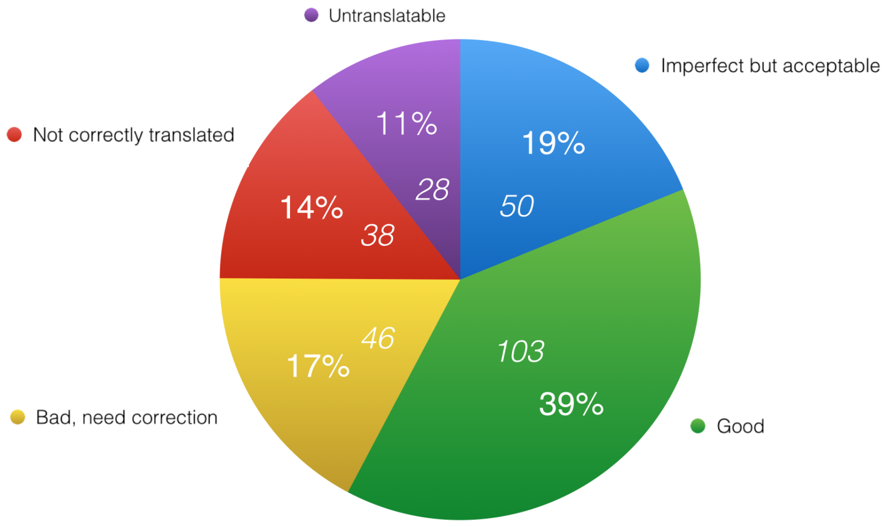


Figure 6.9: Distribution of each translation quality category.

To quantify the performance gap between real scenario performance and current testing performance, the first step is to look at is the distribution of each translation quality category. From Figure 6.9 it can be observed that category "Good", "Imperfect but acceptable" and "Untranslatable" take about 70% of all the translated testing queries. For those three categories, we consider there is no gap between real user input. This is because the "Good" category contains perfect translation while

NDCG		Testing SAME			Testing DIFF			Testing RAW
Experiments		**@1**	**@5**	**@10**	**@1**	**@5**	**@10**	**MRR**
Good	monolingual	0.2174	0.2945	0.3688	0.2350	0.2531	0.3346	0.2374
39%	bilingual	0.1486	0.2451	0.3428	0.2311	0.2624	0.3578	0.2589
Imperfect but acceptable	monolingual	0.2274	0.2968	0.3844	0.2030	0.2564	0.3316	0.2426
19%	bilingual	0.1635	0.2945	0.3777	0.2464	0.2735	0.3331	0.2517
Bad, need correction 17%	monolingual	0.1479	0.2524	0.3487	0.2499	0.2521	0.3446	0.2462
	bilingual	0.0664	0.2283	0.3166	0.1538	0.2220	0.3019	0.1980
	corrected bilingual	0.1723 +160%	0.2817 +23%	0.3463 +9%	0.1137 -26%	0.1957 -12%	0.2646 -12%	0.2046
Not correctly translated 14%	monolingual	0.1714	0.2885	0.3962	0.1912	0.1859	0.3348	0.2154
	bilingual	0.0857	0.2571	0.3749	0.1864	0.2986	0.3934	0.2424
	corrected bilingual	0.0143 -83%	0.2572 0%	0.3122 -16%	0.1388 -25%	0.1742 -7%	0.2983 -24%	0.1556
Untranslatable	monolingual	0.2000	0.2252	0.3346	0.1653	0.2305	0.3227	0.1991
11%	bilingual	0.3051	0.3132	0.4042	0.1128	0.2246	0.3296	0.1960

Table 6.12: Results of different translation categories. Experiments where queries are corrected are compared with the performance of uncorrected version. The percentage of performance change should be estimating the performance gap between real user input testing and our current machine translation testing.

"Imperfect but acceptable" category contains translations with acceptable flaws. Translations in "Imperfect but acceptable" category are the ones might be written differently by English speaker but the current form is also able to find the object user want to search. "Untranslatable" queries are the ones English users won't input because the objects described in those queries are not known by English speakers. Therefore we consider these categories as proper estimations for real scenario input. This distribution indicates that nearly 70% of the machine translated testing queries don't have gap between real user input, which means their testing results can reflect the real performance of the model.

The remaining 30% of the translated testing queries belong to category "Bad, need correction" and "Not correctly translated". We mainly focus on quantifying the performance gap created by these testing queries. Table 6.12 shows all experiment results for testing queries translation quality analysis.

For those two categories, the gap between "bilingual" experiment and "corrected bilingual" gives an estimation for the gap created by these poorly translated queries. It can be seen that in terms of Testing-DIFF, the performance of manual re-translated (corrected) queries is generally worse than that of those machine translated testing queries. A main reason for the gap is because when training and testing queries are translated using the same machine translation system, the translation tend to be similar for the same concept. This lead to a high overlap between the vocabulary of training queries and testing queries. However, for real user inputs, English users might user terms not seen during machine translation, especially for name entities. For example, the French movie "狂爱仪式" has a literal translation of "Mad love ceremony", but its official French name is "Ceremonie d'amour" and its official English name is "Love Rites". A English user is tend to search this target with query "Love Rites" or "Ceremonie d'amour" rather than "Mad love ceremony". Note that term "Ceremonie", "d'amour" and "Rites" are hardly to be encountered when machine translation is used. The first two are French terms therefore can't get from a translation process from Chinese to English. The third is an English term, however it has a more frequently used synonym "ceremony". By observing translation result, we found that machine translation systems tend to choose high frequency terms when there are several equivalent candidates. As a result, real user input tends to introduce more out-of-vocabulary (OOV) terms compared with machine translated queries. The OOV column in Table 6.11 shows the number of OOV terms in each experiment. A term is considered as OOV when it can't be found in our training vocabulary. Note that after manually correction, the number of OOV terms increases dramatically.

Another point worth noticing is that in experiments involved in corrected translations, we also observe improvement occasionally. This illustrates another effect of manually corrected queries: it might bring improvement in translation quality, which can lead to a better ranking result compared with corrupted machine translated queries before correction. However, such effect is unstable and relatively weak compared with the negative effect brought by OOV problem based on the general situation that performance decreases after queries are corrected.

7 Conclusion

For traditional CLIR, dictionary based query translation was "state-of-the-art" CLIR approach. A well-tuned traditional CLIR can be almost as good as a monolingual system in terms of performance[5]. However, building a well functional traditional CLIR system takes a lot of work, like finding proper dictionary, solve coverage problem for phrases and name entities and resolve ambiguity.

This work proposed a neural architecture based CLIR approach, which is much easier to build and more robust to translation errors. Our neural CLIR approach is consist of two parts: bilingual training data and underlying neural ranking model, K-NRM. K-NRM learns translation relationships from bilingual training data by capturing soft-matches from bilingual term pairs and combine soft-matches to generate final score with a set of bins. Each bin has different responsibilities(capture positive relevant evidence or negative relevant evidence) and different importance. This gives the model a lot of freedom in capturing and utilizing translation knowledge as bilingual soft-match signals so that the performance can be optimized, which is considered as the source of effectiveness for neural CLIR. Experiment results show that bilingual training data approach can yield a better performance on K-NRM compared with other query translation based CLIR approaches even given the same translation knowledge source.

Experiments also show that for reranking task, neural CLIR learns more of popularity than relevance signals. This might be caused by a flaw in our experiment methodology: we are currently reranking initial ranking produced by a monolingual system. Most documents for reranking contains the query term indicating a relevance signal, which makes such relevance signal less decisive in ranking. However, this also proves that model in neural CLIR is capable of learning decisive factor in ranking. It leans a combination of relevance signal, which is suppressed by the re-ranking methodology , and popularity, which is discovered by our system here.

Performance of neural CLIR can be further improved by cleaning training dataset by removing ambiguous queries. Also, a rising trend in the curve describing how performance changes with training data size indicates that more training data may also help in improving performance.

Due to the large amount of testing queries, currently English testing queries are generated by Google translation API from Chinese queries, while the real testing scenario should be English user input queries written in English. We did a testing query quality analysis experiment to quantify the affect in performance brought by the gap between machine translated English queries and real user input. Results show that nearly 60% of translated queries are in relatively good quality and won't affect real scenario performance and the remaining 40% queries might result in a 20% gap in performance on average, which is mainly caused by out-of-vocabulary(OOV) problem.

Normally, OOV problem can be mitigated by increasing size of training data so that the vocabulary can be enriched. However, in this case, the OOV problem is caused by a wording difference in machine translation and real user input. For example, a French movie is translated into "Mad love ceremony" by machine translation while real user input is “Love Rites”. The term “Rites” will hardly be seen even if we translate more training data with machine translation. Therefore, increasing training data that translated by machine translation system might not help in solving OOV problem here. Instead, including manually translated queries might help in this case for that vocabularies that can’t be generated in machine translation systems but used by real users can be included.

The next step of this work is to make use of the newly published version of K-NRM that is able to handle phrases to further improve learning of translation relationship [29]. This will give the neural CLIR approach a higher capability in dealing with phrases and name entities, which are big challenges in web search environment.

Bibliography

[1] D. A. Hull and G. Grefenstette, "Querying across languages: A dictionary-based approach to multilingual information retrieval," in *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '96. New York, NY, USA: ACM, 1996, pp. 49–57.

[2] C. Xiong, Z. Dai, J. Callan, Z. Liu, and R. Power, "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*, ser. SIGIR '17. New York, NY, USA: ACM, 2017, pp. 55–64.

[3] I. Vulić and M.-F. Moens, "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*, ser. SIGIR '15. New York, NY, USA: ACM, 2015, pp. 363–372.

[4] C. W. B. Ballesteros Lisa, "Resolving ambiguity for cross-language retrieval," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '98. New York, NY, USA: ACM, 1998, pp. 64–71.

[5] D. W. Oard, "Multilingual information access," in *Encyclopedia of Library and Information Sciencesl.* Oxford, UK: Taylor, Francis, 2010, p. 3682–3687.

[6] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'13. USA: Curran Associates Inc., 2013, pp. 3111–3119.

[7] F. Diaz, B. Mitra, and N. Craswell, "Query expansion with locally-trained word embeddings," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics.* ACL – Association for Computational Linguistics, June 2016.

[8] S. L. Smith, D. H. P. Turban, S. Hamblin, and N. Y. Hammerla, "Offline bilingual word vectors, orthogonal transformations and the inverted softmax," *CoRR*, vol. abs/1702.03859, 2017.

[9] Y. Liu, X. Xie, C. Wang, J.-Y. Nie, M. Zhang, and S. Ma, "Time-aware click model," *ACM Trans. Inf. Syst.*, vol. 35, no. 3, pp. 16:1–16:24, Dec. 2016.

[10] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *TACL*, vol. 5, pp. 135–146, 2017.

[11] D. W. Oard, "A comparative study of query and document translation for cross-language information retrieval," in *Conference of the Association for Machine Translation in the Americas.* Springer, 1998, pp. 472–483.

[12] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, L. Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. S. Corrado, M. Hughes, and J. Dean, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016.

[13] M. Madankar, M. Chandak, and N. Chavhan, "Information retrieval system and machine translation: a review," *Procedia Computer Science*, vol. 78, pp. 845–850, 12 2016.

[14] P. Sujatha and P. Dhavachelvan, "A review on the cross and multilingual information retrieval," *International Journal of Web & Semantic Technology*, vol. 2, no. 4, p. 115, 2011.

[15] H. Nanba, S. Mayumi, and T. Takezawa, "Automatic construction of a bilingual thesaurus using citation analysis," in *Proceedings of the 4th Workshop on Patent Information Retrieval*, ser. PaIR '11. New York, NY, USA: ACM, 2011, pp. 25–30.

[16] D. Demner-Fushman and D. W. Oard, "The effect of bilingual term list size on dictionary-based cross-language information retrieval," in *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on.* IEEE, 2003, pp. 10–pp.

[17] Wikipedia contributors, "Named entity — Wikipedia, the free encyclopedia," 2018, [accessed 6-May-2018]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Named_entity&oldid=818530540.

[18] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Trans. Inf. Syst.*, vol. 20, no. 4, pp. 422–446, Oct. 2002, [accessed 6-May-2018]. [Online]. Available: http://doi.acm.org/10.1145/582415.582418.

[19] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Found. Trends Inf. Retr.*, vol. 3, no. 4, pp. 333–389, Apr. 2009, [accessed 6-May-2018]. [Online]. Available: http://dx.doi.org/10.1561/1500000019.

[20] C. Luo, T. Sakai, Y. Liu, Z. Dou, C. Xiong, and J. Xu, "Overview of the ntcir-13 we want web task," *Proc. NTCIR-13*, 2017.

[21] T. Strohman, D. Metzler, H. Turtle, and W. Croft, "Indri: A language model-based search engine for complex queries," in *Proceedings of the International Conference on Intelligent Analysis*, 2005.

[22] D. M. Christopher, R. Prabhakar, and S. Hinrich, "Introduction to information retrieval," *An Introduction To Information Retrieval*, vol. 151, no. 177, p. 5, 2008.

[23] C. M. Meyer and I. Gurevych, *Wiktionary: A new rival for expert-built lexicons? Exploring the possibilities of collaborative lexicography, chapter 13.* Sylviane Granger and Magali Paquot (Eds.): Electronic Lexicography, Oxford: Oxford University Press, November 2012.

[24] T. Simpson and T. Dao, "Wordnet-based semantic similarity measurement," 2005, [accessed 8-May-2018]. [Online]. Available: https://www.codeproject.com/Articles/11835/WordNet-based-semantic-similarity-measurement0.

[25] N. Rubens, "The application of fuzzy logic to the construction of the ranking function of information retrieval systems," *arXiv preprint cs/0610039*, 2006.

[26] G. Diño, "Baidu translate: The inside story," Mar 2016, [accessed 8-May-2018]. [Online]. Available: https://slator.com/technology/baidu-translate-the-inside-story/.

[27] Google, "Cloud translation api - dynamic translation — google cloud," [accessed 8-May-2018]. [Online]. Available: https://cloud.google.com/translate/.

[28] Z. Dai, C. Xiong, J. Callan, and Z. Liu, "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.* ACM, 2018, pp. 126–134.