Syntactic based approach for grammar question retrieval

Fang, Lanting; Tuan, Luu Anh; Hui, Siu Cheung; Wu, Lenan

2018

Fang, L., Tuan, L. A., Hui, S. C., & Wu, L. (2018). Syntactic based approach for grammar question retrieval. *Information Processing and Management*, 54(2), 184-202.
doi:10.1016/j.ipm.2017.11.004

<https://hdl.handle.net/10356/103976>

<https://doi.org/10.1016/j.ipm.2017.11.004>

© 2017 Elsevier Ltd. All rights reserved. This paper was published in *Information Processing and Management* and is made available with permission of Elsevier Ltd.

Downloaded on 26 Apr 2023 18:24:45 SGT

Syntactic based Approach for Grammar Question Retrieval

Lanting Fang^a, Luu Anh Tuan^b, Siu Cheung Hui^b, Lenan Wu^a

^a*School of Information Science and Engineering, Southeast University, Nanjing, Jiangsu, 210096 CHN*

^b*School of Computer Science and Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798 SG*

Abstract

With the popularity of online education platforms, English learners can learn and practice no matter where they are and what they do. As an important task of learning English, learning English grammar requires students to practice a group of questions targeting on every one of the grammatical focuses. In this paper, we study a novel problem of retrieving questions with similar grammatical focus. Since the grammatical focus similarity is different from textual similarity or sentence syntactic similarity, existing studies cannot be directly applied to our problem. To address this problem, we propose a syntactic based approach for English grammar question retrieval which can capture the grammatical focus of grammar questions effectively. We first propose a new syntactic tree, namely parse-key tree, to capture the English grammar questions' grammatical focus. Then we propose two kernel functions, namely relaxed tree kernel and part-of-speech order kernel, based on which we can compute the similarity between the two parse-key trees. The retrieved grammar questions are ranked according to the similarity between the parse-key trees of query and grammar questions in the collection. If a query is submitted together with answer choices, conceptual similarity and textual similarity are also incorporated to further improve the retrieval accuracy. The performance results have shown that our proposed approach outperforms other state-of-the-art methods.

Keywords: grammar question retrieval, syntactic tree, relaxed tree kernel, POS order kernel

1. Introduction

Online learning has become increasingly popular all around the world over the past few years. For example, in massive open online courses (MOOC), students can watch the lecture videos from top universities around the world and take tests to assess their learning outcome. The rise of e-education has made it possible for people to study what they like no matter where they are and what they do. In year 2015, there were more than 1.5 billion people seeking to learn English [1]. An important task of learning English is to develop a solid foundation in English grammar. As Krashen et al. [2] stated, it is the single most important component of language learning that learners are exposed to massive language input. In order to help the student master a grammar knowledge point, it is important to let the students solve a group of questions targeting on the grammar knowledge point. However, it is tedious and troublesome to find, identify and classify questions that have similar grammatical focus manually. As such, it is highly desirable to develop a system for retrieving English questions with similar grammatical focus for students to practice and learn English grammar.

Multiple choice question (MCQ) is one of the most frequent form used in educational testing [3], and they are widely used to evaluate the understanding of English grammar. A MCQ question consists of a stem (i.e. a sentence with a blank space) and four choices. Only one of the choices is the correct answer to the question stem. In this paper, we therefore investigate the problem of retrieving English grammar questions with similar grammatical focus by discussing English grammar MCQs, though our proposed approach can be easily extended to retrieve other kinds of grammar questions. Specifically, given a collection of grammar MCQs, and the query grammar MCQ submitted by the user, we aim to retrieve the grammar MCQs from the collection such that the retrieved MCQs have similar grammatical focus to the MCQ query. The grammatical focus of a MCQ is the grammatical knowledge point assessed by the MCQ. The grammatical knowledge

point assessed by the MCQ is captured by the grammatical concepts of the blank space in the question stem and grammatical structure around the blank space, where the grammatical concepts of the blank space is the grammatical categories of the words that should be filled in the blank space e.g., comparison-of-adjective and relative-pronoun. In this paper, we consider two kinds of MCQ queries: partial grammar MCQ and complete grammar MCQ. We will give a detail description of MCQ queries in Section 3.

In information retrieval (IR), there are many existing models such as statistical analysis approaches [4, 5, 6] and syntactic analysis approaches [7, 8, 9] which are widely used for text and sentence retrieval. Such models are mostly based on statistical analysis of term occurrences in textual content or syntactic analysis of sentences. These models have been shown to be effective for many practical applications such as web search engine and question-answering system. However, we cannot apply these models to our grammar MCQ retrieval problem. The statistical analysis approaches only capture the textual content, which cannot reflect the grammatical focus of the MCQ. Moreover, though the syntactic analysis approaches can capture the grammatical structure information and many focus extraction techniques are proposed to capture the focus of a query, the focus of a grammar MCQ is different from the focus of a general sentence. The existing focus extraction and syntactic analysis approaches can not fill well in MCQ retrieval.

To capture the grammatical focus of the MCQ, we propose a syntactic tree structure according to the grammatical structure of the MCQ and position of the blank space in the question stem. For the query MCQ and each MCQ in the collection, we compute the similarity between their syntactic tree structures. For a partial grammar MCQ, we rank the MCQs in the collection according to the similarity between their syntactic tree structures. Then based on an assumption that two grammar MCQs are irrelevant if the words in the blank space have different grammatical concepts, we use a grammar checker to filter out irrelevant MCQs for a partial grammar MCQ. In addition, we summarize a list of common grammatical concepts that appear in English tests like GaoKao. For example, words “better”, “best” and “worst” all belong to the grammatical concept of comparison-of-adjective. For a complete MCQ, we can determine its grammatical concepts based on its choices and then compute the conceptual similarity between two MCQs. We also compute the textual similarity between the choices of two MCQs. The conceptual and textual similarities are incorporated to further enhance the retrieval performance.

The contribution of our study is summarized as follows: (1) We propose a new syntactic tree, namely parse-key tree. The proposed parse-key tree can capture the grammatical focus of MCQ. (2) We propose two kernel functions, namely relaxed tree kernel and part-of-speech (POS) order kernel to evaluate the similarity between two parse-key trees. Specifically, the relaxed tree kernel captures the number of common substructures between two parse-key trees while the POS order kernel captures the POS position similarity between two parse-key trees. (3) We propose to use a grammar checker to filter out irrelevant MCQs for partial grammar MCQ queries. In addition, we summarize a list of common grammatical concepts that appear in English tests to further enhance the retrieval performance for complete grammar MCQ queries.

The rest of the paper is organized as follows. Section 2 discusses the related work on question retrieval. Section 3 presents the objective of English grammar question retrieval. Section 4 presents the proposed syntactic based approach for grammar question retrieval. Section 5 presents the experimental setup including dataset collection, statistics information and performance metrics. Section 6 gives the performance evaluation of the proposed approach. Finally, Section 7 concludes the paper.

2. Related Work

MCQ plays a major role in educational testing systems. Banerjee et al. [10] present a MCQ answering system that reads a single document and identifies the correct answer to a question from a set of possible answer options. Majumde et al. [11] propose a sentence selection method for MCQ generation. Agarwal et al. [12] propose a system that automatically generates questions from natural language text using discourse connectives. In this paper, we focus on grammar MCQ retrieval system that finds MCQs in the collection with similar grammatical focus to the input query.

A class of commonly used retrieval techniques are statistical analysis approaches. The statistical analysis approaches mostly rely on term occurrences in the documents to find related documents. Some classical

methods such as TF-IDF [5], BM25 [6], language modeling methods [13] like maximum likelihood estimation, and many others, all belong to this category.

Another class of commonly used retrieval techniques are syntactical analysis approaches. They can be classified into two categories: part-of-speech (POS) and dependency relation feature based approaches and syntactic tree based approaches.

POS and dependency relation feature based approaches [14, 15] exploit the POS and dependency relation to improve the retrieval results. Mohammad et al. [16] compute the text overlap similarity based on syntactic and semantic features. Shah and Croft [17] use POS tags to find the headword in a given query to help find the focus of the query. Barr et al. [18] apply POS tags to Web search queries to improve the search results. Recently, Wang et al. [19] incorporate POS into bag-of-word and Markov Random Field to strengthen the conventional IR models for biomedical information retrieval. Schubotz et al. [7] use POS based distance methods to extract identifiers to improve math information retrieval results. These research works have shown that by comparing the POS tags or pattern of the words between the query and documents, the retrieval accuracy can be improved. David et al. [20] use both statistical features and syntactic features for information retrieval. Park et al. [21] compute the relevance between the query and document by matching the different types of syntactic relations in their dependency parsed trees. Bendersky et al. [22] propose a retrieval framework that models dependencies between arbitrary concepts in the query rather than just query terms. In [23], syntactic roles of words are used to detect key concepts to help rank the candidate questions. Maxwell and Croft [24] show that the retrieval performance can be improved through the use of dependency parsing techniques in the extraction of non-adjacent subsets of dependent terms. Recently, Chinkina et al. [25] develop an online information retrieval system that retrieves Web documents based on the grammatical structures they contain.

Different from the statistical and syntactical approaches, we focus on grammar MCQ retrieval problem, which aims to find grammar MCQs with similar grammatical focus. The statistical approaches only consider the text content. Although the syntactical approaches use the POS tags and dependency relations in a sentence, they do not consider the structure information of syntactic roles.

Recently, there are some other syntactical approaches exploit tree structural information. Those approaches extracted information from the syntactic trees parsed from the sentences. Moschitti and Basili [26] take advantage of tree kernels to generate large feature set from the syntactic parse trees for non-factoid answers classification. Lin et al. [27] propose a tree similarity measurement for classifying questions by represent the parse trees of questions as multi-dimensional sequences. Yao et al. [28] extract features from a tree edit distance model for aligning an answer sentence tree with the question tree. Galitsky et al. [29] propose a syntactic parse tree based approach to learn syntactic information at the level of paragraphs. Tymoshenko et al. [30] extensively investigate the use of syntactic and semantic structures enriched with LD semantics. Higashinaka et al. [8] propose a syntactic filter based on dependency tree to ascertain valid sentences. Those methods extract query focus, i.e. the intent of the query, based on the Name Entity Recognizer (NER) types ¹ and noun phrases. However, the existing focus extraction methods can not be used in the grammar MCQ retrieval since term “focus” in MCQ retrieval problem has a totally different meaning from the “focus” in question answering system. For example, “*What is the best way to get to Singapore zoo*” is similar to “*How to get to Singapore zoo*” in sentence retrieval because both of the two sentence aim to ask the way to Singapore zoo. However, they are not similar in MCQ retrieval. In MCQ retrieval, the focus is its grammatical focus (i.e., the knowledge point assessed by the MCQ). In the aforementioned example, the focus of the first sentence is “comparison-of-adjective”, while the focus of the second sentence is “question-word”. In fact, “*What is the best way to get to Singapore zoo*” is more similar to “*What is the best way to get to Santiago Bernabu Stadium*” in MCQ retrieval. However, they are not similar in sentence retrieval. To solve this issue, we propose a parse-key tree based method to extract the grammatical focus of a grammar MCQ.

Finally, a preliminary version of this article was published in [31]. Even though an effective solution is presented in that previous work, the solution is designed for partial MCQ query only. Specifically, the query

¹NER labels sequences of words in a text which are the names of things, such as Location, Person, Organization, Money, Percent, Date, Time and Lexical relations etc.

Table 1: Sample English grammar questions

Q_1	Question Stem: <i>The waitress * we thought deserves a Service Quality award has resigned.</i> Choices: A. <i>whom</i> B. <i>where</i> C. <i>which</i> D. <i>when</i> Answer: A. <i>whom</i>
Q_2	Question Stem: <i>My favorite teacher is my English teacher, and she is by far the * teacher that I have ever had.</i> Choices: A. <i>good</i> B. <i>better</i> C. <i>best</i> D. <i>worst</i> Answer: B. <i>best</i>
Q_3	Question Stem: <i>The man speaking * stage is her respected physical professor.</i> Choices: A. <i>in</i> B. <i>at</i> C. <i>on</i> D. <i>outside</i> Answer: C. <i>on</i>

only contains the question stem. Here in this long version of the article, we extend the grammar MCQ retrieval problem by considering two types of query: partial MCQ query and complete MCQ query. We also propose three new methods to extend the existing parse-key tree based solution to tackle the two types of query. More specifically, in order to address the partial MCQ query, besides the parse-key tree proposed in [31], we propose a new strategy to further improve the performance. The strategy is based on an assumption that two grammar MCQs are irrelevant if the words in the blank space have different grammatical concepts. In this strategy, we use a grammar check to prune some irrelevant MCQs in the collection (to be explained in Section 4.6.1). For a complete MCQ query, the words that should be filled in the blank is highly related to the grammatical focus. Thus, to utilize such information, we first summarize a list of common grammatical concepts, and propose a conceptual similarity (to be presented in Section 4.5.1). We also take the textual similarity into consideration (to be presented in Section 4.5.2). The conceptual and textual similarity can further enhance the retrieval performance.

3. English Grammar Question Retrieval

A MCQ question consists of a stem (i.e. a sentence with a blank space) and four choices. Only one of the choices is the correct answer to the question stem. Table 1 gives three examples of English grammar MCQs. In the question stem, we use the symbol ‘*’ to denote the blank space, which should be filled in with the correct choice. For example, the question stem of Q_1 in Table 1 is “The waitress * we thought deserves a Service Quality award has resigned.”, choice A “whom” is the correct answer. The completed sentence filled by correct answer is “The waitress whom we thought deserves a Service Quality award has resigned.”.

In this paper, we investigate the problem of retrieving grammar MCQs with similar grammatical focus for students to practice their grammar knowledge. In this paper, we consider two different kinds of grammar MCQ queries: partial grammar MCQ (i.e. grammar MCQ with the question stem only) and complete grammar MCQ (i.e. grammar MCQ with the question stem and the choices). An example partial grammar MCQ query is given as follows:

Q_p : *The lady * you saw performing on stage is our favorite English teacher.*

An example of complete MCQ query is given as follows:

Q_c : *The lady * you saw performing on stage is our favorite English teacher.*

Choices: A. *who* B. *which* C. *whom* D. *whose*

Answer: C. *whom*

The objective of grammar MCQ retrieval problem is to find grammar MCQs with similar grammatical focus. Note that the grammatical focus similarity is different from the textual similarity or sentence syntactic similarity. The retrieved grammar MCQs should have similar grammatical knowledge point, i.e. the concepts of the blank in the stem and grammatical structure around the blank, to the grammar MCQ query. Consider the MCQ query Q_p and the three grammar MCQs in Table 1. Grammar MCQ Q_2 is more similar to Q_p

Figure 1: Proposed approach

textually since they share many common terms. Meanwhile, in terms of sentence syntactic similarity, grammar MCQ Q_3 is more similar to Q_p since they share similar sentence structure. However, Q_1 is the one with the similar grammatical focus to Q_p . Specifically, Q_1 and Q_p have similar grammatical concept and grammatical structure around the blank, but Q_2 and Q_3 do not. Q_2 and Q_p are not similar on both grammatical focus and grammatical structure around the blank. Although Q_3 and Q_p have similar grammatical structure, Q_3 and Q_p are not similar. Because Q_1 and Q_p both focus on the grammatical concept of “clause”, but Q_3 focuses on the grammatical concept of “preposition”. As such, if we only use the text retrieval or sentence retrieval method, we will not be able to compute the grammatical focus similarity between the query and MCQs in the collection effectively. In this paper, we propose a parse-key tree based method for the purpose of grammar MCQ retrieval.

4. Proposed approach

In this section, we introduce our proposed approach to address the grammar MCQ retrieval problem.

4.1. Overview

Figure 1 shows the framework of our proposed approach for grammar MCQ retrieval. We consider two types of query grammar MCQs: partial grammar MCQ (grammar MCQ with the question stem only) and complete grammar MCQ (grammar MCQ with question stem and choices). For a partial query grammar MCQ, we first construct the parse-key trees for the query grammar MCQ and each grammar MCQ in the collection from their question stems. The parse-key tree can capture the grammatical focus of the grammar MCQ. Then we compute the similarity between the parse-key trees of the query and each grammar MCQ in the collection based on relaxed tree kernel and POS order kernel. The computed similarity will be passed to Query Retrieval process to rank all grammar MCQs. In addition, we use a grammar checker to filter out irrelevant questions for partial grammar MCQ query.

For a complete query grammar MCQ, we can further improve the performance by utilizing the additional answer choices information. Specifically, the Question Reconstruction process will insert the correct answer into the blank space to complete the question stem. Different from the partial query grammar MCQ, we use the completed question stem to construct the parse-key tree. Moreover, based on the correct answer, we can determine a set of concepts based on a list of rules summarized by experts. We compute the conceptual similarity and textual similarity between the correct answer of two grammar MCQs. The conceptual and textual similarity will be passed to the Query Retrieval process to enhance the performance.

In the following, we will present the details of each process in the framework.

4.2. Question Reconstruction

Recall that a complete grammar MCQ query contains the answer choices information. Thus, when user submits a complete grammar MCQ query, the Question Reconstruction process will complete the question stem for both the query grammar MCQ and the grammar MCQs in the collection. Specifically, for the query

grammar MCQ and each grammar MCQ in the collection, we fill in the blank with the correct answer to reconstruct the question stem.

Consider the example of complete grammar MCQ query in Section 1. Choice C “whom” is the correct answer. The reconstructed question stem will be “The lady whom you saw performing on stage is our favorite English teacher.”.

4.3. Parse-key Tree Construction

As mentioned in Section 2, the existing focus extraction methods can not be used in grammar MCQ retrieval. To capture the focus of a grammar MCQ, we propose a new structure named parse-key tree based on syntactic parse tree. The proposed parse-key tree uses the sub-tree extracted from the syntactic tree, which avoid the noise information caused by the entire tree. Since the position of the blank space in the question stem may provide hint on the grammatical focus of the query, we enriched the blank space information to the extracted sub-tree.

Before introducing the parse-key tree, we first present *syntactic parsing* and *word-blank distance extraction*, which will be used to define the parse-key tree.

4.3.1. Syntactic Parsing

Syntactic parsing is used to capture the syntactic information from a grammar question through the parse tree obtained by using the Stanford parser [32] to parse the question. A parse tree is an ordered, rooted syntactic tree that represents the grammatical structure of a sentence. Figure 2 shows the parse tree of the example partial query Q_p “The lady * you saw performing on stage is our favorite English teacher.”. Some abbreviations used in Figure 2 are shown in Table 2. The leaf nodes in Figure 2 are the words in the sample query. The leaf nodes’ parents (i.e. pre-terminal nodes) are their part-of-speech (POS) tags, e.g. noun, verb, pronoun, etc.

4.3.2. Word-blank Distance Extraction

The position of the blank space (“*”) in the question stem may provide hint on the grammatical focus of the query. For example, the question stem “The lady * you saw performing on stage is our favorite English teacher.” has the focus on grammatical property of “clause”, while the question stem “The lady whom you saw performing * stage is our favorite English teacher.” has the focus on grammatical property of “preposition”.

To identify the grammatical focus of an English grammar question, we define word-blank distance to capture the relative position of each word to the blank space in the question.

Definition (Word-blank Distance). *Given a grammar question stem with a blank space, the word-blank distance of each word in the question is defined as the distance between the positions of the word and of blank space.*

Specifically, we use 0 to represent the word-blank distance of the blank space, i to represent the word-blank distance of the i -th word after the blank space, and $-j$ to represent the word-blank distance of j -th word before the blank space. For example, for the question stem “lady * you saw”, we can generate the following word-blank distance information: {“lady”:-1, “*”:0, “you”: 1, “saw”:2}. “lady” is the first word before the blank space, so its word-blank distance is -1. “you” is the first word after the blank space, so its word-blank distance is 1.

4.3.3. Parse-key Tree Construction

The syntactic parse tree can be used to represent the grammatical structure of a grammar MCQ. However, it is difficult to find the grammatical focus of a grammar MCQ if we use the entire tree. Therefore, we propose *parse-key tree* to represent the grammatical focus of a grammar MCQ. A *parse-key tree* is a sub-parse tree extracted from the parse tree of the grammar MCQ incorporated with the word-blank distance. Before discussing parse-key tree, we first give the definition on neighbor nodes as follows:

Table 2: Some Abbreviations used in Parse Tree

Abbreviations	Meaning
S	Sentence
NP	Noun phrase
VP	Verb phrase
SBAR	Subordinate clause
NN	Noun, singular or mass
SYM	Symbol
PRP	Personal pronoun
VBD	Verb, past tense

Figure 2: Parse tree for the sample query

Definition (Neighbor Nodes) *Given a parse tree and two integer parameters M and N , we define neighbor nodes as a set of leaf nodes with word-blank distance in the range $[M, N]$.*

Consider the example of partial query Q_p “The lady * you saw performing on stage is our favorite English teacher.” whose syntactic parse tree is shown in Figure 2. With the parameters $M = -1$ and $N = 2$, we can extract the neighbor nodes {“lady”, “*”, “you”, “saw”}.

Finally, we are now ready to present the definition of parse-key tree.

Definition (Parse-key Tree). *Given a grammar question with a blank space, the parse-key tree is defined as a sub-parse tree, which consists of the neighbor nodes and their predecessors, together with the information on word-blank distance.*

Algorithm 1 shows the Parse-key Tree Construction process. The algorithm takes input as a question stem Q_{stem} and returns the parse-key tree T_{PK} . First, it uses the Stanford parser to parse the question stem to obtain the parse tree (line 1). Next, it computes the word-blank distance for the words in Q_{stem} (line 2). Then, it constructs a parse-key tree from the parse tree and word-blank distances (lines 3-7). To do this, it first extracts the neighbor nodes from the parse tree and the predecessor nodes of all neighbor nodes. For example, the predecessor nodes of the neighbor node “lady” are {“S”, “NP”, “NP”, “NN”}. The parse-key tree T_{PK} is then initialized as the sub-parse tree consisting of neighbor nodes and their predecessors. The sub-parse tree enclosed by the dotted line in Figure 2 is the sub-parse tree with parameters $M = -1$ and $N = 2$ of the parse tree. Then, for each leaf node n in T_{PK} , we add a sibling node labeled with n ’s word-blank distance to n . For example, word-blank distance of the “lady” node is “-1”, we add a sibling node labeled “-1” to “lady”. Finally, the parse-key tree T_{PK} is returned. The parse-key tree of the example partial

Algorithm 1: Parse-key Tree Construction

Input: Q_{stem} : A question stem
Output: T_{PK} : A parse-key tree

- 1 $T \leftarrow$ parse tree obtained from Q_{stem}
- 2 Compute word-blank distances for the words in Q_{stem}
- 3 $neighbor \leftarrow$ neighbor nodes extracted from T
- 4 $predecessor \leftarrow$ predecessor nodes of neighbor nodes extracted from T
- 5 $T_{PK} \leftarrow$ subtree consisting of nodes in $neighbor$ and $predecessor$
- 6 **for** each leaf node n in T_{PK} **do**
- 7 Add sibling node labeled with n 's word-blank distance to n
- 8 **return** T_{PK}

Figure 3: Parse-key tree for the sample partial query with $M = -1$ and $N = 2$

grammar MCQ query Q_p is shown in Figure 3. If the submitted query also contains answer choices, the correct answer will be substituted into the blank space of the query, and the parse-key tree of the complete grammar MCQ query Q_d will be constructed as shown in Figure 4. As illustrated in Figure 3 and Figure 4, the parent nodes of the leaf nodes with word-blank distance 0 in partial grammar MCQ query (i.e. “*”) and complete grammar MCQ query (i.e. “whom”) are “SYM” and “WP” respectively.

4.4. Relaxed Tree Kernel and POS order Kernel

In the Parse-key Tree Construction process, we construct the parse-key trees of the query and each grammar MCQ in the collection to capture both the grammatical structure and properties that they focus on. In this section, we introduce two kernel functions, named relaxed tree kernel and POS order kernel to compute the similarity between two parse-key trees.

4.4.1. Relaxed Tree Kernel

Tree kernel [33] is a method to compute the number of common substructures between two trees [30, 34, 35, 36]. It can be used to capture the important relational information between two trees. In an English grammar question, the punctuations (e.g. ‘?’, ‘;’) can change the grammatical structure of a question. Meanwhile, the word-blank distance of the word may provide hint on the grammatical focus of a query. Thus punctuation nodes and word-blank distance nodes have an significant impact on the grammatical focus. However, the tree kernel ignores such information, and thus it does not fit well in our problem. To address this problem, we propose a new kernel function, named relaxed tree kernel.

The main idea behind relaxed tree kernel is to calculate the number of common tree fragments between two parse-key trees. Before introducing the definition of tree fragments, we first define some terminology which will be used later.

Definition (Production). *Given a node, the production at the node is the relationship between the node and its children. It is denoted as “ $node \rightarrow child_1 \dots child_n$ ”, where $child_1 \dots child_n$ are the children of the node.*

Figure 4: Parse-key tree for the sample complete query with $M = -1$ and $N = 2$

Figure 5: An example for pre-terminal nodes

For example, consider the node “SBAR” in Figure 3. The children of “SBAR” are “SYM” and “S”, so its production is “SBAR $\rightarrow$ SYM S”.

Note that the punctuation nodes can change the grammatical structure of a question, while word-blank distance nodes may provide hint on the grammatical focus of a query. Therefore, we define specific production to enhance the effect of such nodes.

Definition (Specific Production). *Given a pre-terminal node (i.e. node that only has leaf node) called PT-node, a specific production at PT-node is the relationship between PT-node and one of its children. It is denoted as “PT-node $\rightarrow$ child”, where child is either a punctuation node, e.g. ‘,’ or a word-blank distance node, e.g. ‘-1’. The specific production is empty if none of PT-node’s children is a punctuation node or a word-blank distance node. Two specific productions are the same only if they are equal and not empty.*

For example, consider the example in Figure 5(a). The children of the node “,” are “,” and “-1”. So it has two specific productions: (1) “ , $\rightarrow$, ” and (2) “ , $\rightarrow$ -1”. Consider the example in Figure 5(b). The children of the node “PRP” are “you” and “1”, so its specific production is “ PRP $\rightarrow$ 1”.

The definition of tree fragment is given as follows:

Definition (Tree Fragment). *Given a parse-key tree, a tree fragment is a connected sub-graph such that (1) it contains more than one node, and (2) it is a union of productions.*

Consider the parse-key tree shown in Figure 6(a). It has three different tree fragments, which are given in Figure 6(b). Note that “PRP $\rightarrow$ you” is not a tree fragment, because “PRP $\rightarrow$ you” is not the a production.

The relaxed tree kernel between two parse-key trees is based on the number of common tree fragments. The common tree fragments are defined as follows:

Definition (Common Tree Fragments). *Given two tree fragments F_1 and F_2 , we say F_1 and F_2 are common tree fragment if and only if there exists a bijection f between the node sets of the two tree, such that for any node $u \in F_1$ and its corresponding node $f(u) \in F_2$, either their production or their specific production are the same.*

According to the definition of Tree Fragment, a tree fragment is a subgraph of the parse-key tree. Thus, the number of common tree fragments between the two trees can effectively capture their relational

Figure 6: An example for tree fragment

information. Next, we discuss how to compute the relaxed tree kernel, which is the total number of the common tree fragments between two parse-key trees.

The main idea is that we can compute the relaxed tree kernel for a pair of parse-key trees recursively. The pseudocode is reported in Algorithm 2. The algorithm takes input as two parse-key trees T_q and T_d , and the output of the algorithm is the total number of common tree fragments in the two trees. We use N_q and N_d to denote the set of non-leaf nodes in T_q and T_d , respectively (lines 1–2). The relaxed tree kernel K_{TPK} is initialized as 0 (line 3). For each node $n_q \in N_q$ and node $n_d \in N_d$, we compute the number of common tree fragments rooted at n_q and n_d , denoted by $CTF(n_q, n_d)$, and add the number to K_{TPK} (lines 3–6).

The function $CTF(n_q, n_d)$ takes nodes n_q and n_d as input and computes the number of common tree fragments rooted at n_q and n_d as output. The $CTF(n_q, n_d)$ is defined as follows:

1. If the production and the specific production at n_q and n_d are both different, then $CTF(n_q, n_d) = 0$.
2. If the production or one specific production at n_q and n_d are the same, and n_q and n_d are pre-terminal nodes, then $CTF(n_q, n_d) = 1$.
3. Otherwise, if the productions at n_q and n_d are the same, and n_q and n_d are not pre-terminal nodes, then

$$CTF(n_q, n_d) = \prod_{j=1}^{nc(n_q)} [1 + CTF(ch(n_q, j), ch(n_d, j))]$$

where $nc(n_q)$ is the number of children of n_q and $ch(n_q, j)$ is the j -th child of the node n_q .

We have relaxed the condition on computing the number of common tree fragments as compared to the tree kernel proposed in [33]. The original tree kernel function requires two corresponding nodes must have the same production, while in our function, two corresponding nodes can either have the same production or the same specific production.

In the calculation of CTF , it is quite straight-forward for the first two cases. For the third case, CTF is defined recursively. Note that to get a common tree fragment rooted at n_q and n_d , we can either simply taking the productions at n_q and n_d , or together the productions at n_q and n_d with taking any one of the common tree fragments rooted at the children of n_q and n_d . Thus, there are $(1 + CTF(ch(n_q, j), ch(n_d, j)))$ choices for the j -th child. Putting them together, we can see that CTF can correctly compute the number of common tree fragments rooted at n_q and n_d .

The normalized relaxed tree kernel is defined as:

$$S_{TPK}(T_q, T_d) = \frac{K_{TPK}(T_q, T_d) - K_{min}}{K_{max} - K_{min}} \quad (1)$$

where K_{max} and K_{min} are the maximum and the minimum values of the relaxed tree kernel between the query and each of the questions in the collection, respectively.

Algorithm 2: Relaxed Tree Kernel

Input: T_q : parse-key tree of query
 T_d : parse-key tree of question in collection
Output: K_{TPK} : Relaxed Tree Kernel

```
1  $N_q \leftarrow$  all the nodes in  $T_q$  (exclude leaf nodes)
2  $N_d \leftarrow$  all the nodes in  $T_d$  (exclude leaf nodes)
3  $K_{TPK} = 0$ 
4 for  $n_q \in N_q$  do
5   | for  $n_d \in N_d$  do
6   |   |  $K_{TPK} = K_{TPK} + CTF(n_q, n_d)$ 
7 return  $K_{TPK}$ ;
```

4.4.2. POS Order Kernel

POS order kernel computes the POS position similarity between two parse-key trees. To compute the POS order kernel, we first give the definition of POS tags list and the longest common sub-POS sequence (LCS sequence) between two parse-key trees.

Definition (POS tags list). *Given a parse-key tree, the POS tags list of the parse-key tree is the list of its pre-terminal nodes' labels.*

Consider the parse-key tree in Figure 3, the POS tags list of this parse-key tree is {"NN", "SYM", "PRP", "VBD"}.

Definition (LCS sequence). *Given two parse-key trees, the longest common sub-POS sequence (LCS sequence) between the two parse-key trees is the longest common subsequence between their POS tags lists.*

Consider two POS tags list, {"NN", "SYM", "PRP", "VBD"} and {"VBD", "JJ", "NN", "PRP"}. The LCS sequence between the two POS tags list is {"NN", "PRP"}.

Given two parse-key trees T_q and T_d , let LS be their LCS sequence, and L_q and L_d be their POS tags list, respectively. Let s be a POS tag in LS . The order similarity of s in LS is defined as:

$$O(s) = \frac{1}{|P_{L_q}(s) - P_{L_d}(s)| + 1} \quad (2)$$

where $P_{L_q}(s)$ is the position of the POS tag s in L_q , and $P_{L_d}(s)$ is the position of the POS tag s in L_d .

For example, the position distance of "NN" between $L_q = \{"NN", "SYM", "PRP", "VBD"\}$ and $L_d = \{"VBD", "JJ", "NN", "PRP"\}$ is $|P_{L_q}("NN") - P_{L_d}("NN")| = 2$, where $P_{L_q}("NN") = 1$, $P_{L_d}("NN") = 3$.

The POS order kernel POS_{order} between T_q and T_d is defined as the summation of order similarity of all elements in their LCS sequence divided by the longer length of L_q and L_d i.e.,

$$POS_{order}(T_q, T_d) = \frac{1}{\max(|L_q|, |L_d|)} \sum_{s \in LS} O(s) \quad (3)$$

where $|L_q|$ and $|L_d|$ are the lengths of lists L_q and L_d , respectively..

The normalized POS order kernel is defined as:

$$S_o(T_q, T_d) = \frac{POS_{order}(T_q, T_d) - POS_{min}}{POS_{max} - POS_{min}} \quad (4)$$

where POS_{max} and POS_{min} are the maximum and the minimum values of the POS order kernel between the query and each of the questions in the collection, respectively.

4.5. Conceptual and Textual Similarity

Recall that when the user submits a complete grammar MCQ query, the additional choices information can be utilized to further improve the performance. In this subsection, we present how to incorporate such

Table 3: Example concepts and words

Category	Example Concept Words
relative-pronoun	which, who, where, whom, etc.
comparison-of-adjective	best, worst, most, better, worse, etc.
question-word	what, who, whom, whose, etc.
clause	how, that, who, what, whom, etc.

Table 4: Conceptual categories for query and questions in Table 1

Question	Conceptual Categories
Query	{“relative-pronoun”, “clause”, “question-word”}
Q_1	{“relative-pronoun”, “clause”, “question-word”}
Q_2	{“comparison-of-adjectives”}
Q_3	{“preposition-of-time”, “preposition-of-position”}

information to enhance the retrieval performance. Specifically, for the choices of two grammar MCQs, we compute the conceptual similarity and textual similarity.

4.5.1. Conceptual Similarity

In English tests, there are some words that always have the same grammatical properties. For example, words such as “better”, “best” and “worst” all have the grammatical concepts of “comparison-of-adjective”. Therefore, we build a list of common grammatical concepts that appear in English tests. Each concept contains a set of words which belong to this grammatical category. In total, there are 76 such concepts created manually. Table 3 gives some examples of conceptual categories and part of their words list. Given a grammar MCQ, we can determine its conceptual categories based on whether the correct answer are in the concept words list. If yes, it extracts the corresponding concept name. The conceptual category of a grammar MCQ will be $\{\emptyset\}$ if the words at the blank space are not found in the concept words list. For the example complete grammar MCQ query, the correct answer C : *whom* is in the concept words list of “relative-pronoun”, “clause” and “question-word”. Therefore, the query’s conceptual categories are {“relative-pronoun”, “clause”, “question-word”}. Table 4 shows the conceptual categories of the query and grammar MCQs in Table 1. As shown in Table 4, the conceptual categories of the query are more similar to Q_1 than Q_2 or Q_3 .

The conceptual similarity score between grammar MCQ query Q_q and grammar MCQ in collection Q_d is defined as:

$$S_c(Q_q, Q_d) = \frac{|C_q \cap C_d|}{|C_q \cup C_d|} \quad (5)$$

where C_q and C_d are conceptual categories of Q_q and Q_d . If C_q and C_d are both empty, we set the value of $S_c(Q_q, Q_d)$ to 1.

4.5.2. Textual Similarity

Apart from conceptual similarity, we also extract textual similarity between the choices of two grammar MCQs, i.e., we only focus on the correct answer and extract it as textual feature. Table 5 shows the textual features of the example query and questions. As shown in Table 5, the textual features for the query are {“whom”}, and the textual features for Q_1 , Q_2 and Q_3 are {“whom”}, {“best”} and {“on”} respectively.

The textual similarity score between grammar MCQ query Q_q and grammar MCQ in collection Q_d is defined as:

$$S_t(Q_q, Q_d) = \frac{|E_q \cap E_d|}{|E_q \cup E_d|} \quad (6)$$

where E_q and E_d are textual features of Q_q and Q_d . If E_q and E_d are both empty, we set the value of $S_t(Q_q, Q_d)$ to 1.

Table 5: Textual Features for query and questions in Table 1

Question	Textual Features
Query	{“whom”}
Q_1	{“whom”}
Q_2	{“best”}
Q_3	{“on”}

4.6. Query Retrieval

In this section, we discuss how to retrieve grammar MCQs with similar grammatical focus for partial grammar MCQ queries and complete grammar MCQ queries, respectively.

4.6.1. Partial Query

For partial grammar MCQ query, the grammatical focus similarity between the query question Q_q and each question Q_d in the collection is calculated based on relaxed tree kernel and POS order kernel. In the experiments which follow, we combined the relaxed tree kernel and POS order kernel with RBF kernel-based SVM-regression. After computing the similarity scores between the query question and all questions in the collection, we rank the grammar MCQs in the collection according to their similarity scores. Then, we use a grammar checker to filter out irrelevant questions.

The grammatical focus of an English grammar question is related to the expected answer in the blank space of the query. Therefore, we assume that the query and questions in the collection will be irrelevant if the words in their blank space have different grammatical concepts. Based on this assumption, after computing the parse-key tree similarity between the query and questions in the collection, we use a grammar checker² to check the grammatical structure in the sentences constructed from the query and the choices of each question in the retrieval result in order to filter out unrelated questions. Specifically, for every related grammar MCQ appeared in the retrieval result, we fill in the blank space of the question stem with each of the four choices. We will then get four reconstructed question stem sentences. The four reconstructed sentences are then passed to the grammar checker to check their grammatical validity. If all of the query sentences are unable to pass the grammar validity check, the question will be removed from the ranked retrieval result. That means if any of the reconstructed queries can pass the grammar validity check, the question will be retained in the retrieval result.

4.6.2. Complete Query

The grammatical focus similarity between the query question Q_q and each question Q_d in the collection is calculated based on relaxed tree kernel, POS order kernel, conceptual similarity and textual similarity. Next, we combined the sub-kernels and sub-similarities with RBF kernel-based SVM-regression. Then we rank the grammar MCQs in the collection according to their similarity scores.

Note that in the aforementioned approach, we assume that the correct answer of the complete query grammar MCQ is already known. But our approach can also be easily extended to handle the situation when users do not know the correct answer. Specifically, we can repeat the process four times. One choice is regarded as the correct answer at a time. Then for each grammar MCQ in the collection, we have computed four combined scores for it, each of the score corresponds to a choice. The highest combined score is selected to be the final score for the grammar MCQ. Grammar MCQs in the collection are ranked by this final score.

5. Experimental Setup

In the experiments, we use a total of 4,580 English grammar MCQ questions as the dataset for performance evaluation. The questions are assembled from primary school leaving examination (PSLE) and

²<https://languagetool.org>

Table 6: Concept tag statistics in dataset

Concept	Pr	Concept	Pr
relative-pronoun	9.8	question-word	9.5
possessive-adjective	7.2	quantifier	7.2
personal-pronoun	5.5	articles	5.2
preposition-of-time	5.0	clause	4.5
indefinite-pronoun	4.5	modal-auxiliary	4.0

Table 7: POS statistics in dataset

Part of Speech	Pr	Part of Speech	Pr
preposition	14.3	pronoun, personal	9.8
determiner	9.5	noun, singular	9.3
verb, past tense	9.0	verb, 3rd person singular present	6.5
verb, non-3rd person singular present	6.5	model	6.3
adverb	4.3	Wh-determiner	3.5

Gaokao (NCEE). PSLE is an annual national examination in Singapore administered by the Ministry of Education and taken by all students near the end of their sixth year in primary school before moving on to secondary school. Gaokao is an academic examination held annually in China, and is a prerequisite for entrance into almost all higher education institutions at the undergraduate level. We extract 400 grammar MCQs as the query and the remained 4180 MCQs will be served as questions in the collection for query retrieval. The extraction is done randomly taking care that the grammatical concept distribution remained proportional. Figure 7 shows the question stem length distributions of grammar MCQs in the evaluation data. The average length of the question stems is 16.36 words.

The relevance between each query and grammar MCQ in the collection pair is manually annotated on a 5-level relevance scale (“bad”, “fair”, “good”, “excellent”, and “perfect”), where “bad” represent the grammar MCQs in the collection is irrelevant to the query and “perfect” represent the grammar MCQ in the collection is the most relevant to the query. The annotation is done by 10 native English speakers. In the experiments, we consider relevance level “bad” as irrelevant pair and the others as relevant pair. For each query MCQ, about 12.9% of the MCQs in the collection are relevant on average. We use 200 queries for training and 200 for testing. For each training query, we use 5 MCQs in the collection labeled with 5 different scales for training our regression model.

Figure 7: % of each length in the data set. Label 25+ refers to grammar MCQ of length 25 and above.

As mentioned above, we have a total of 76 concept categories, for each of the possible concept category we collect all grammar MCQs in whose answer at least one word is in the concept category. We report the percentage of such grammar MCQs in our collection, denoted by Pr . For easier reading, we only show the top 10 frequent concept categories in Table 6. There is a total of 47% grammar MCQs contain at least one concept category. We also report the percentage of grammar MCQ answers that at least one word is tagged with the POS tag. The top 10 frequent POS tags is shown in Table 7

In the experiments, we use precision-recall curve and the following three metrics for evaluation:

- Precision at rank K ($P@K$): It measures the percentage of relevant questions ranked at top K positions and is calculated as follows:

$$P@K = \frac{1}{N_q} \sum_{q=1}^{N_q} \frac{Relevant(q)}{K}$$

where N_q is the number of queries and $Relevant(q)$ is the number of relevant questions at top K results of query q .

- Mean Average Precision ³ at rank K ($MAP@K$): It measures the mean of the average precision scores for each query and is calculated as follows:

$$MAP@K = \frac{1}{N_q} \sum_{q=1}^{N_q} AveP(q)$$

where N_q is the number of queries and $AveP(q)$ is the average precision score of query q at top K results.

- Mean reciprocal rank ⁴ (MRR): It measures the average of the reciprocal ranks of results for a sample of queries and is calculated as follows:

$$MRR = \frac{1}{N_q} \sum_{q=1}^{N_q} \frac{1}{rank_q}$$

where N_q is the number of queries and $rank_q$ is the rank position of the first relevant document for the query q .

In our experiments, we evaluate the performance of the methods using the metrics $P@1$, $P@3$, $P@5$, $MAP@20$ and MRR.

5.1. Methods Used for Performance Comparison

In our experimental study, we compare the performance of the following methods (S_{TP} , S_{TPK} and S_o are models proposed in this paper.):

- BM25 [6]: This is the text retrieval method, which is based on statistical analysis of term occurrences in the text.
- phRank [24]: This is the retrieval model based on the term ranking algorithm. We use the parameters mentioned in paper [24].
- Syntactic analysis [20]: This method considers both statistical features and syntactic features. Then this method use online SVMRank to rank the questions. We use the mentioned training set to train the ranking model.

³<https://www.kaggle.com/wiki/MeanAveragePrecision>

⁴https://en.wikipedia.org/wiki/Mean_reciprocal_rank

Figure 8: Precision-recall curve for partial grammar MCQ query

Figure 9: Precision-recall curve for complete grammar MCQ query

- Tree kernel [33]: This is the method based on syntactic parse tree with the original tree kernel function for counting common sub-trees between syntactic trees.
- REL tree [30]: This method uses tree structure enriched with focus-based relations and lexical relations. This method uses the tree similarity and a list of feature vectors to rank the questions. We use the mentioned training set to train the ranking model.
- S_{TP} : This is the model using entire tree structure enriched with word-blank distance to represent the grammar MCQ. The relaxed tree kernel is used as the ranking model.
- S_{TPK} : This is the model using the parse-key tree to represent the grammar MCQ. The tree kernel is used as the ranking model.
- S_o : This is the model using the parse-key tree to represent the grammar MCQ. The POS order kernel is used as the ranking model.
- S_c : This is the model using the conceptual similarity as the ranking model.
- S_t : This is the model using the textual similarity as the ranking model.
- Proposed: This is the model using $S_{TPK} + S_o$ + grammar check for partial grammar MCQ query and using $S_{TPK} + S_o + S_c + S_t$ for complete grammar MCQ query.

6. Performance Evaluation

In this section, we discuss the performance evaluation of the proposed approach

6.1. Parameter Tuning and Learning

We use regressions with 3-fold cross-validation to model the similarity score between each query and collection question pair. Firstly, we vary the value of M and N to construct the parse-key tree. Specifically, the value of M is taken from $[-5, 0]$, while the value of N is from $[0, 5]$. Then we use RBF-kernel based SVM-regression to build the similarity score model. We vary the penalty factor $C \in \{2^{-5}, 2^{-3}, \dots, 2^{13}, 2^{15}\}$, single parameter $\gamma \in \{2^{-15}, 2^{-13}, \dots, 2^1, 2^3\}$ and epsilon-tube $\epsilon \in \{2^{-15}, 2^{-13}, \dots, 2^1, 2^3\}$. According to the cross-validation results, we set the parameters $M = -1$ and $N = 2$.

6.2. Performance with respect to Partial Grammar MCQ Query

In this subsection, we evaluate the performance of our proposed approach with respect to the partial grammar MCQ query. We report the performance results in Table 8. Table 9 reports the comparison of

Table 8: Performance results for partial grammar MCQ query. Statistical significance with $p < 0.01$ against all state-of-the-art models is marked with †.

Models	P@1	P@3	P@5	MAP@20	MRR
State-of-the-art					
BM25	29.2	16.7	17.5	31.4	41.0
phRank	26.3	15.8	12.6	24.0	37.7
Syntactic analysis	29.2	20.8	19.2	31.1	42.2
Tree kernel	41.7	33.3	25.8	37.7	54.6
REL tree	26.1	17.4	15.7	26.0	37.0
Our Individual Models					
S_{TP}	45.8	34.7	35.0	42.1	56.7
S_{TPK}	54.2	47.2†	43.3†	54.8†	64.6
S_o	50.0	36.1†	33.3	47.8	61.2
Our Model Combinations					
$S_{TPK} + S_o$	70.8†	48.6†	43.3†	57.9†	79.1†
$S_{TPK} + S_o$ + grammar checker (Proposed)	75.0†	48.6†	43.3†	59.1†	81.2†

different evaluated methods with respect to the MRR value. For example, the value in 2 row and 3 column is the ratio of MRR of method phRank and Syntactic analysis.

Comparison with state-of-the-art methods. We compare the precision-recall curves of the proposed method with BM25, phRank, Tree kernel, Syntactic analysis and REL tree in Figure 8. From the results, we can see the proposed method is better than all of the state-of-the-art methods.

We also show the performance comparison with other evaluation metrics for the proposed approach and the state-of-the-art methods in Table 8.

We can observe that our proposed approach improve the performance by around 25.8%(P@5)-45.8%(P@1) when compared with BM25. One intuitive explanation is that the purpose of grammar question retrieval is to find questions with most similar grammatical focus. Text term based retrieval methods are not effective for this task. phRank is a retrieval model based on the term ranking algorithm. However, the terms selected by phRank are not the focus terms for grammar questions. We illustrate the effect of syntactic information with an example. Consider a partial grammar MCQ query “*The exclusion of the star players from the final match between the long-time rivals * been identified as the reason for the dismal performance.*” (Answer: *has*) from our evaluation set. The BM25 returns the highest ranked grammar MCQ “*The performance of the boys and the girls at the concert * special mention.*” (Answer: *deserves*). The phRank returns the highest ranked grammar MCQ “*For * reason would anyone want to queue up when they can apply online?*” (Answer: *what*). Yet, the two questions are not relevant to the query based on their grammatical focus. In our evaluation set, the grammar MCQ “*Neither of the managers * turned up for the meeting. We have to wait for them.*” (Answer: *has*) is “perfect” relevant to the partial grammar MCQ query. This grammar MCQ is ranked the first by our method.

Table 8 also shows that Syntactic analysis methods do not fit in well with grammar MCQ retrieval problem. Syntactic analysis method uses the POS and syntactic roles as the syntactic features. Without order information for POS and structural information for syntactic roles, this method cannot retrieves questions with similar grammatical focus effectively. For example, given a partial grammar MCQ query “*He’s not a bad guy; we really * give him another chance.*” (Answer: *should*). The top ranked grammar MCQ returned by syntactic Analysis methods is “*For him * stage is just * means of making a living.*” (Answer: *the...a*). The two grammar MCQs have many similar syntactic elements like the occurrences of POS tags. However, they are not relevant since they are not similar because their grammatical concepts at the blank space are different. Our method rank the grammar MCQ “*In crowded places like airports and railway stations, you * take care of your luggage.*” (Answer: *must*) at the top place. This grammar MCQ is

Table 9: Comparison of different evaluated methods with respect to the MRR value for partial grammar MCQ query. The value in the table is the result of row method MRR score divided by the column method MRR score.

	BM25	phRank	Syntactic analysis	Tree kernel	REL tree	S_{TP}	S_{TPK}	S_o	$S_{TPK} + S_o$	Proposed
BM25	1.00	1.09	0.97	0.75	1.11	0.72	0.63	0.67	0.52	0.50
phRank	0.92	1.00	0.89	0.69	1.02	0.66	0.58	0.62	0.48	0.46
Syntactic analysis	1.03	1.12	1.00	0.77	1.14	0.74	0.65	0.69	0.53	0.52
Tree kernel	1.33	1.45	1.29	1.00	1.48	0.96	0.85	0.89	0.69	0.67
REL tree	0.90	0.98	0.88	0.68	1.00	0.65	0.57	0.60	0.47	0.46
S_{TP}	1.38	1.50	1.34	1.04	1.53	1.00	0.88	0.93	0.72	0.70
S_{TPK}	1.58	1.71	1.53	1.18	1.75	1.14	1.00	1.06	0.82	0.80
S_o	1.49	1.62	1.45	1.12	1.65	1.08	0.95	1.00	0.77	0.75
$S_{TPK} + S_o$	1.93	2.10	1.87	1.45	2.14	1.40	1.22	1.29	1.00	0.97
Proposed	1.98	2.15	1.92	1.49	2.19	1.43	1.26	1.33	1.03	1.00

“excellent” similar to the query since both of them focus on the grammatical concept of “modal-auxiliary” and their grammatical structures around the blank space are similar.

When compared with Tree kernel and REL tree methods, our proposed approach improves the performance by 15.3%(P@3)-33.3%(P@1) and 27.6%(P@5)-48.9%(P@1) for all metrics. It shows that our proposed parse-key tree is more effective in capturing the focus of grammar MCQs than syntactic parse tree. Additionally, the traditional focus extraction method is not useful in grammar question retrieval. For example, given the sample partial grammar MCQ query “*The lady * you saw performing on stage is our favorite English teacher.*” (Answer: *whom*). The proposed method returns the top ranked grammar MCQ “*The waitress * we thought deserves a Service Quality award has resigned.*” (Answer: *whom*). This grammar MCQ is “perfect” relevant to the sample query since the two grammar MCQs are similar in grammatical structure around the blank space and they both focus on the grammatical concept of “clause”. However, the Tree kernel method returns the top ranked grammar MCQ “*Which of the three ways shall I take to the village? * way as you please.*” (Answer: *Any*). The REL tree returns the top ranked grammar MCQ “*The mayor has offered a reward of \$5000 to * who can capture the tiger alive or dead.*” (Answer: *anyone*). Although the two grammar MCQs are a little similar with the query based on their grammatical structure, both the two grammar MCQs are not relevant to the sample query since they focus on the grammatical concept of “general-determiner” and “indefinite-pronoun”, respectively

As illustrated in the performance results, our approach achieves better performance than other methods because the proposed parse-key tree can effectively capture the grammatical structure and properties of the grammar questions and is able to rank the grammar questions based on both relaxed tree kernel and POS order kernel.

Effect of Ranking Models. In this set of experiments, we study the effectiveness of individual models.

As shown in Table 8, S_{TPK} improves 7.9%(MRR)-12.7%(MAP@20) compared with S_{TP} . This result shows that using sub-tree to construct parse-key tree is more suitable to represent the grammar MCQ compared with entire parse tree. The use of $S_{TPK} + S_o$ improves the performance by 0%(P@5)-16.6(P@1) and 10%(P@5)-19/2(P@1) compared with using S_{TPK} and S_o , respectively. This is because relaxed tree kernel capture the number of common substructures between two parse-key trees while the POS order kernel capture the POS position similarity between two parse-key trees. Both kernel measures are useful for grammar question retrieval.

Effect of Grammar Checker. We next investigate the effect of the grammar checker. Recall that as discussed in Section 4.6.1, we use a grammar checker to check the grammatical structure in order to filter out unrelated questions. From Table 8 we observe that the use of grammar checker for filtering out irrelevant

Table 10: Performance results for complete grammar MCQ query. Statistical significance with $p < 0.01$ against all state-of-the-art models is marked with †

Models	P@1	P@3	P@5	MAP@20	MRR
State-of-the-art					
BM25	16.7	16.7	17.5	29.9	34.3
phRank	31.3	20.8	18.8	26.8	41.6
Syntactic analysis	29.1	20.8	20.0	31.3	42.3
Tree kernel	20.8	12.5	17.5	27.7	37.5
REL tree	33.3	18.1	15.0	26.7	42.5
Our Individual Models					
S_{TP}	45.8	48.6†	37.5	43.1	61.7
S_{TPK}	87.5†	69.4†	61.7†	70.8†	91.3†
S_o	66.7†	63.9†	53.3†	60.9†	74.1†
S_c	45.8	50.0†	50.0†	51.7	50.7
S_t	73.3†	60.0†	48.7†	72.2†	75.9†
Our Model Combination					
$S_{TPK}+S_o$	83.3†	72.2†	65.0†	71.8†	89.4†
$S_{TPK}+S_o+S_c$	91.7†	83.3†	74.2†	77.4†	95.1†
$S_{TPK}+S_o+S_c+S_t$ (Proposed)	100.0†	91.7†	86.7†	86.9†	100.0†

questions from the retrieval results can improve the performance by up to 4.2% for P@1 when compared to the counterpart without grammar checker.

6.3. Performance with respect to Complete grammar MCQ Query

In this subsection, we evaluate the performance of our proposed approach and the-state-of-the-art methods with respect to the complete grammar MCQ query. We first evaluate the overall performance of the methods. Then we study how the ranking models affect the performance of our approach. Table 10 shows the performance results of the Complete grammar MCQ query. Table 11 shows the comparison of different evaluated methods with respect to the MRR value for complete grammar MCQ query.

Comparison with State-of-the-art Methods. We first compare our proposed approach with other state-of-the-art methods. Figure 9 shows the precision-recall curves compared with state-of-the-art methods. It can be seen in Figure 9, the propose method is better than all of the state-of-the-art methods. It can be seen in Table 10, our proposed approach performs significantly better than all other comparison methods for all metrics. Generally, our proposed approach improves 57.0%(MAP@20)-83.3%(P@1), 58.4%(MRR)-70.9%(P@3), 55.6%(MAP@20)-70.9%(P@1), 59.3%(MAP@20)-79.2%(P@1,P@2) and 57.5%(MRR)-73.6%(P@3) when compared to BM25, phRank, Syntactic analysis, Tree kernel and REL tree, respectively. It shows that our proposed approach is more effective than other methods in the task of grammar question retrieval.

Effects of Ranking Models. We next study effectiveness of ranking models. Table 10 shows that using parse-key tree (S_{TPK}) to represent the grammar MCQ improves 20.8%(P@2)-41.7%(P@1) compared with using parse tree enriched with word-blank distance (S_{TP}).

Similar to the case for the partial grammar MCQ query, we observe that the performance of the combination of S_{TPK} and S_o in the proposed approach is better than that of the individual S_{TPK} or S_o .

We study the effectiveness of the conceptual similarity (S_c) and textual similarity (S_t). From Table 10 we observe that using $S_{TPK}+S_o+S_c$ is able to improve the performance compared with using $S_{TPK}+S_o$. It shows that the conceptual similarity is useful for improving the retrieval performance. The combination of all individual models in the proposed approach have achieved the best overall performance result.

Table 11: Comparison of different evaluated methods with respect to the MRR value for complete grammar MCQ query. The value in the table is the result of row method MRR score divided by the column method MRR score.

	BM25	phRank	Syntactic analysis	Tree kernel	REL tree	S_{TP}	S_{TPK}	S_o	S_c	S_t	$S_{TPK} + S_o$	$S_{TPK} + S_o + S_c$	Proposed
BM25	1.00	0.82	0.81	0.91	0.81	0.56	0.38	0.46	0.68	0.45	0.38	0.36	0.34
phRank	1.21	1.00	0.98	1.11	0.98	0.67	0.46	0.56	0.82	0.55	0.47	0.44	0.42
Syntactic analysis	1.23	1.02	1.00	1.13	1.00	0.69	0.46	0.57	0.83	0.56	0.47	0.44	0.42
Tree kernel	1.09	0.90	0.89	1.00	0.88	0.61	0.41	0.51	0.74	0.49	0.42	0.39	0.38
REL tree	1.24	1.02	1.00	1.13	1.00	0.69	0.47	0.57	0.84	0.56	0.48	0.45	0.42
S_{TP}	1.80	1.48	1.46	1.65	1.45	1.00	0.68	0.83	1.22	0.81	0.69	0.65	0.62
S_{TPK}	2.66	2.19	2.16	2.43	2.15	1.48	1.00	1.23	1.80	1.20	1.02	0.96	0.91
S_o	2.16	1.78	1.75	1.98	1.74	1.20	0.81	1.00	1.46	0.98	0.83	0.78	0.74
S_c	1.48	1.22	1.20	1.35	1.19	0.82	0.56	0.68	1.00	0.67	0.57	0.53	0.51
S_t	2.21	1.82	1.79	2.02	1.79	1.23	0.83	1.02	1.50	1.00	0.85	0.80	0.76
$S_{TPK} + S_o$	2.61	2.15	2.11	2.38	2.10	1.45	0.98	1.21	1.76	1.18	1.00	0.94	0.89
$S_{TPK} + S_o + S_c$	2.77	2.29	2.25	2.54	2.24	1.54	1.04	1.28	1.88	1.25	1.06	1.00	0.95
Proposed	2.92	2.40	2.36	2.67	2.35	1.62	1.10	1.35	1.97	1.32	1.12	1.05	1.00

7. Conclusion

In this paper, we proposed an effective syntactic based approach for English grammar questions retrieval. The input English grammar question query can be a partial grammar MCQ query or a complete grammar MCQ query. In the proposed approach, we proposed a parse-key tree to capture the grammatical focus of the grammar MCQ based on its syntactic tree and position of the blank space. The MCQs in the collection are ranked based on relaxed tree kernel and POS order kernel. If the submitted query is a complete grammar MCQ query, the retrieval process can further be improved by using the conceptual and textual similarities between the query and MCQs in the collection. We conducted extensive experiments to show that our proposed approach significantly outperforms statistical retrieval methods such as BM25 and other syntactical methods for grammar MCQ retrieval.

The proposed approach can be used in English learning system (e.g. MOOC) for English learners to search English grammar questions. The technology proposed in this paper may be applicable to compute syntactic similarity for other areas of information retrieval. In the future, we plan to incorporate user's interests (like user's level and goal) in our retrieval system.

Acknowledgement

The work is supported by the Research Innovation Program for College Graduates of Jiangsu Province (Grant No. KYLX15-0076).

References

- [1] R. Noack and L. Gamio, "The worlds languages, in 7 maps and charts," *The Washington Post*, vol. 4, no. 23, pp. 65–70, 2015.
- [2] S. Krashen, "Some issues relating to the monitor model," *On Tesol*, vol. 77, pp. 144–158, 1977.
- [3] A. M. Brady, "Assessment of learning with multiple-choice questions," *Nurse Education in Practice*, vol. 5, no. 4, pp. 238–242, 2005.
- [4] J. Vilares, M. A. Alonso, Y. Doval, and M. Vilares, "Studying the effect and treatment of misspelled queries in cross-language information retrieval," *Information Processing and Management*, vol. 52, no. 4, pp. 646–657, 2016.
- [5] S. E. Robertson, S. Walker, M. Beaulieu, M. Gatford, and A. Payne, "Okapi at trec-4," *Nist Special Publication Sp*, pp. 73–96, 1996.
- [6] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.
- [7] M. Schubotz, A. Grigorev, M. Leich, H. S. Cohl, N. Meuschke, B. Gipp, A. S. Youssef, and V. Markl, "Semantification of identifiers in mathematics for better math information retrieval," in *Proceedings of the 39th international ACM SIGIR conference on Research and development in information retrieval*, 2016, pp. 135–144.

- [8] R. Higashinaka, N. Kobayashi, T. Hirano, C. Miyazaki, T. Meguro, T. Makino, and Y. Matsuo, "Syntactic filtering and content-based retrieval of twitter sentences for the generation of system utterances in dialogue systems," in *Situated Dialog in Speech-Based Human-Computer Interaction*, 2016, pp. 15–26.
- [9] R. S. Roy, S. Agarwal, N. Ganguly, and M. Choudhury, "Syntactic complexity of web search queries through the lenses of language models, networks and users," *Information Processing and Management*, vol. 52, no. 5, pp. 923–948, 2016.
- [10] S. Banerjee, P. Bhaskar, P. Pakray, S. Bandyopadhyay, and A. F. Gelbukh, "Multiple choice question (mcq) answering system for entrance examination." in *CLEF 2013 Working Notes*, 2013.
- [11] M. Majumder and S. K. Saha, "A system for generating multiple choice questions: With a novel approach for sentence selection," 2015, p. 64712.
- [12] M. Agarwal, R. Shah, and P. Mannem, "Automatic question generation using discourse cues," in *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, 2011, pp. 1–9.
- [13] V. Lavrenko and W. B. Croft, "Relevance based language models," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 120–127.
- [14] W. Zhang, Z. Ming, Y. Zhang, L. Nie, T. Liu, and T. Chua, "The use of dependency relation graph to enhance the term weighting in question retrieval," in *Proceedings of the 25th international conference on Computational linguistics: Technical papers*, 2012, pp. 3105–3120.
- [15] W. Song, M. Feng, N. Gu, and L. Wenying, "Question similarity calculation for faq answering," in *Third International Conference on Semantics, Knowledge and Grid*, 2007, pp. 298–301.
- [16] A. S. Mohammad, Z. Jaradat, A. A. Mahmoud, and Y. Jararweh, "Paraphrase identification and semantic text similarity analysis in arabic news tweets using lexical, syntactic, and semantic features," *Information Processing and Management*, vol. 53, no. 3, pp. 640–652, 2017.
- [17] C. Shah and W. B. Croft, "Evaluating high accuracy retrieval techniques," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, 2004, pp. 2–9.
- [18] C. Barr, R. Jones, and M. Regelson, "The linguistic structure of english web-search queries," in *Proceedings of the conference on Empirical methods in natural language processing*, 2008, pp. 1021–1030.
- [19] Y. Wang, S. Wu, D. Li, S. Mehrabi, and H. Liu, "A part-of-speech term weighting scheme for biomedical information retrieval," *Journal of Biomedical Informatics*, 2016.
- [20] D. Carmel, A. Mejer, Y. Pinter, and I. Szpektor, "Improving term weighting for community question answering search using syntactic analysis," in *Proceedings of the 23th ACM international conference on Information and knowledge management*, 2014, pp. 351–360.
- [21] J. H. Park, W. B. Croft, and D. A. Smith, "A quasi-synchronous dependence model for information retrieval," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, 2011, pp. 17–26.
- [22] M. Bendersky and W. B. Croft, "Modeling higher-order term dependencies in information retrieval using query hyper-graphs," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, 2012, pp. 941–950.
- [23] W. Zhang, Z. Ming, Y. Zhang, T. Liu, and T. S. Chua, "Exploring key concept paraphrasing based on pivot language translation for question retrieval." in *Proceedings of the 29th conference on Association for the advancement of artificial intelligence*, 2015, pp. 410–416.
- [24] K. T. Maxwell and W. B. Croft, "Compact query term selection using topically related text," in *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, 2013, pp. 583–592.
- [25] M. Chinkina, M. Kannan, and D. Meurers, "Online information retrieval for language learning," in *Proceedings of the 54th annual meeting of the Association for computational linguistics*, 2016, pp. 7–12.
- [26] A. Moschitti and R. Basili, "A tree kernel approach to question and answer classification in question answering systems," in *Proceedings of the 5th International Conference on Language Resources and Evaluation*, 2006, pp. 1–510.
- [27] Z. Lin, H. Wang, and S. McClean, "Tree similarity measurement for classifying questions by syntactic structures," in *International Conference on Intelligent Computing*, 2016, pp. 379–390.
- [28] X. Yao, B. Van Durme, C. Callison-Burch, and P. Clark, "Answer extraction as sequence tagging with tree edit distance," in *North American Chapter of the Association for Computational Linguistics*, 2013.
- [29] B. Galitsky, D. I. Ilvovsky, S. O. Kuznetsov, and F. Strok, "Matching sets of parse trees for answering multi-sentence questions." in *Proceedings of Recent Advances in Natural Language Processing*, 2013, pp. 285–293.
- [30] K. Tymoshenko and A. Moschitti, "Assessing the impact of syntactic and semantic structures for answer passages reranking," in *Proceedings of the 24th ACM international conference on Information and knowledge management*, 2015, pp. 1451–1460.
- [31] L. Fang, L. A. Tuan, L. Wu, and S. C. Hui, "A syntactic parse-key tree-based approach for english grammar question retrieval," in *International Conference on Applications of Natural Language to Information Systems*, 2017, pp. 353–365.
- [32] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks." in *Proceedings of the conference on Empirical methods in natural language processing*, 2014, pp. 740–750.
- [33] M. Collins and N. Duffy, "Convolution kernels for natural language," in *Advances in Neural Information Processing Systems*, 2001, pp. 625–632.
- [34] K. Wang, Z. Ming, and T. S. Chua, "A syntactic tree matching approach to finding similar questions in community-based qa services," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 2009, pp. 187–194.
- [35] S. Bloehdorn and A. Moschitti, "Structure and semantics for expressive text kernels," in *Proceedings of the 16th ACM conference on Information and knowledge management*, 2007, pp. 861–864.
- [36] A. Moschitti, "Efficient convolution kernels for dependency and constituent syntactic trees," in *European conference on*

Machine learning, 2006, pp. 318–329.