Personalized Question Recommendation for English Grammar Learning

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

Learning English grammar is a very challenging task for many students especially for non-native English speakers. To learn English well, it is important to understand the concepts of English grammar with lots of practice on exercise questions. Previous recommendation systems for learning English mainly focused on recommending reading materials and vocabulary. Different from reading material and vocabulary recommendations, grammar question recommendation should recommend questions that have similar grammatical structure and usage to the question of interest. The content similarity calculation methods used in existing recommendation methods cannot represent the similarity between grammar questions effectively. In this paper, we propose a content-based approach for personalized grammar question recommendation which recommends similar grammatical structure and usage questions for further practicing. Specifically, we propose a novel structure named parse-key tree to capture the grammatical structure and usage of grammar questions. We then propose three measures to compute the similarity between the question query and database questions for grammar question recommendation. Additionally, we incorporated the proposed recommendation method into a Web-based English grammar learning system and presented its performance evaluation in this paper. The experimental results have shown that the proposed approach outperforms other classical and state-of-the-art methods in recommending relevant grammar questions.

Keywords: content-based recommendation, grammar question recommendation, syntactic parse-key tree

1. Introduction

English, which is widely used for international communications, is probably one of the most important languages nowadays. To learn English well, it is important to understand the concepts of English grammar with lots of practice on exercise questions according to grammatical usage such as pronouns, prepositions, etc. Most of the current grammar learning systems provide the learning of grammar concepts with description and video material, and the related exercises according to grammar topics. For further practicing, students are usually interested in finding questions with similar grammatical structures and usage, especially for those grammar topics in which they are weak. Finding such questions manually are both tedious and troublesome. Therefore, it is highly useful to provide personalized grammar question recommendation to recommend relevant grammar questions according to user preference and needs.

Currently, a number of recommendation systems has been proposed for learning English such as reading materials recommendation Hsu et al. (2013); Lo et al. (2013) and vocabulary recommendation Chen and Chung (2008); Huang et al. (2012). These proposed systems have used content-based and collaborative filtering approaches for their implementation. In this research, we focus on English grammar question recommendation based on multiple choice questions (MCQs), which are the most common form of English grammar questions. A MCQ grammar question consists of two parts: a stem which refers to a question with a blank space, and four choices for users to select the correct answer. Different from reading material and vocabulary recommendations, grammar question recommendation should recommend questions that have similar grammatical structure and usage according to the question of interest. The recommendation methods used in existing recommendation systems (e.g. reading material and vocabulary) cannot fit well in grammar questions recommendation.

There are many existing models such as statistical analysis approach Robertson and Zaragoza (2009); Zhang et al. (2014b,c) and syntactic analysis approach Lease (2007); Zhang et al. (2012); Song et al. (2008) which are widely used for calculating the similarity between text and sentence. 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, recommending grammar questions based on text and sentence similarity is not effective as it only captures similar textual content or syntactic sentence structure between the query question and database questions. It does not reflect their similarity according to the grammatical structure and usage in the context of grammar questions.

In this paper, we propose a content-based approach to recommend grammar questions that have similar grammatical structural and usage to user's interest. In the proposed approach, a novel syntactic tree structure, called parse-key tree, is used to capture the grammatical structure and usage of grammar questions. The parse-key tree similarity as well as the conceptual and textual similarity of the question query and database questions are then computed for grammar question recommendation. The proposed grammar question recommendation approach has been incorporated into a Web-based English grammar learning system.

To the best of our knowledge, our work is the first attempt to provide personalized grammar question recommendation system to recommend grammar questions that have similar grammatical structural and usage to user's interest. The major contributions of the work presented in this paper are as follows:

- We proposed a novel syntactic content-based approach for personalised grammar question recommendation.
- We proposed parse-key tree similarity as well as conceptual similarity and textual similarity to capture the syntactic content similarity between grammar questions.
- The proposed grammar question recommendation approach has been incorporated into a Web-based English grammar learning system.

The rest of the paper is organized as follows. Section 2 reviews the related work on recommendation systems for English reading materials and vocabulary. Section 3 discusses grammar question recommendation and reviews the related techniques. Section 4 gives an overview of the system architecture of our English grammar learning system with question recommendation. Section 5 presents the proposed grammar question recommendation approach. Section 6 gives the performance evaluation. Finally, Section 7 concludes the paper.

2. Related Work

There are mainly three types of recommendation systems Thorat et al. (2015), namely content-based systems, collaborative filtering systems and hybrid systems. The content-based recommendation systems generate recommendations based on the profile of user's preference and the item's description Lops et al. (2011). Huang et al. (2015) proposed a neural probabilistic model for automatic citation recommendation. Li et al. (2012) proposed a self-adjusting e-course generation method which composes personalized e-courses meeting individual learners demands. The collaborative filtering recommendation systems generate recommendations based on a large amount of information on other similar users' behaviors and preference Liu et al. (2012). Ding et al. (2015) proposed a mining model based on convolution neural network for identifying whether the user has a consumption intention, so that better tailored products or services can be recommended. Seo et al. (2017) proposed a recommendation system based on friendship strength in social network services. The hybrid recommendation systems combine the two aforementioned techniques Barragáns-Martínez et al. (2010). Qian et al. (2014) combined personal interest, interpersonal interest similarity, and interpersonal influence to unify personalized recommendation model based on probabilistic matrix factorization. In this section, we introduce the related works of online learning recommendation systems Damiano et al. (2015); Santos and Boticario (2015); Dwivedi and Bharadwaj (2015). In particular, we focus on reviewing some of the online recommendation systems for English reading material and vocabulary.

Hsu (2008) developed an online personalized English learning recommendation system to provide students with reading lessons that suit their interests in order to increase their motivation to learn. The system is based on the analysis of students' preferences (i.e. reading behaviors) using content-based analysis, collaborative filtering and data mining techniques. Hsu et al. (2010) developed an English reading material recommendation system using a knowledge engineering approach. The system provides English reading recommendation by taking both preferences and knowledge of individual students as well as categories and traits of articles into consideration. In addition, Hsu et al. (2013) also proposed a personalized mobile approach for reading material recommendation. It gathers students' reading preferences on topics and guides students to read articles that match their preferences and knowledge levels. Different from the above approaches, Lo et al. (2013) proposed an online annotation system for EFL reading comprehension. The system allows readers to analyze paragraphs of text on web pages and provides annotation tools to add personal ideas to the highlighted text.

Besides reading comprehension, vocabulary knowledge is also one of the most important components in learning English language. To learn vocabulary well, students always need to memorize a large number of vocabulary. Chen and Chung (2008) proposed a personalized mobile English vocabulary learning system based on Item Response Theory (IRT) and learning memory cycles. The proposed system recommends appropriate English vocabulary for learners according to their ability, difficulty parameters of the learned vocabulary and the test results. Huang et al. (2012) developed a ubiquitous English vocabulary learning system to support a systematic vocabulary learning process using the near-synonyms and similar-looking (NSSL) technology. In addition, video clips are also used as the learning material. The technology is able to adapt to the real-world contexts of students and recommend appropriate learning material to them to learn new vocabulary.

Apart from reading material and vocabulary, English

grammar is another important component in learning English. Grammar question recommendation aims to recommend grammar questions with similar grammatical structures and usage. Currently, there is not much related work proposed on grammar question recommendation. In addition, the current techniques on reading material and vocabulary recommendation are also not suitable for grammar question recommendation. In this paper, we propose a content-based approach to recommend relevant grammar questions to help students practice and understand better on grammar concepts in an effective way.

3. Grammar Question Retrieval and Recommendation

Table 1: Sample English Grammar Questions

Q1	Stem: *The waitress ___ we thought deserves a Service Quality award has resigned.* Choices: *A. whom B. where C. which D. when* Answer: *A. whom*
Q2	Stem: *My favorite teacher is my English teacher, and she is by far the ___ teacher that I have ever had.* Choices: *A. good B. better C. best D. worst* Answer: *B. best*

In the database, each MCQ grammar question consists of a stem (i.e. a sentence with a blank or answer position) and four choices. Associated with each question, it also contains the correct answer. Table 1 gives two sample English grammar questions. In grammar question recommendation, it recommends relevant questions from the database which are similar to the query question according to grammatical structure and usage. Note that the meaning of similarity here is referring to not only textual similarity but also the similarity in grammatical structure and usage to the query question.

For example, given the following sample query question:

Stem: *The lady ___ you saw performing on stage is our favorite English teacher.*
Choices: *A. who B. which C. whom D. whose*
Answer: C. whom

It is more similar to question Q1 than question Q2 as both of them are about asking the grammatical usage on clause. Even though the sample query question and question Q2 share many similar words, they are in fact not similar due to the fact that question Q2 is about the grammatical usage on adjective according to the blank or answer position.

In information retrieval, there are mainly two types of techniques for text and sentence retrieval: statistical analysis approach and syntactic analysis approach. The statistical analysis approach Wang et al. (2016a); Zhang et al. (2014a); Wang et al. (2017a) mostly relies on term occurrences in the documents to find related documents. Some classical methods such as TF-IDF Robertson et al. (1996) and BM25 Robertson and Zaragoza (2009) follow this approach. The syntactic analysis approach exploits the linguistic information for improving the retrieval results. In particular, part-of-speech (POS) of words in sentences is commonly used in this approach. Barr et al. (2008) applied POS tags to Web search queries to improve the search results. Wang et al. (2016b) incorporated POS into bag-of-words and Markov Random Field to strengthen the conventional IR models for biomedical information retrieval. Schubotz et al. (2016) used 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.

Apart from POS tagging, there are some other syntactic analysis approaches which exploit the dependency structures of sentences. Carmel et al. (2014) used both statistical features and syntactic features for information retrieval. Park et al. (2011) computed the relevancy between the query and document by matching the different types of syntactic relations in their dependency parsed trees. Maxwell and Croft (2013) showed 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. (2016) developed an online information retrieval system that retrieves Web documents based on the grammatical structures they contain. In addition to POS tagging and dependency relations, some other approaches also investigated the use of syntactic tree to help retrieve relevant sentences. Wang et al. (2009) proposed a retrieval framework based on syntactic tree structure to find similar questions in a community based question-answering system. Higashinaka et al. (2016) proposed a syntactic filter based on dependency tree to ascertain valid sentences.

However, for grammar question recommendation, it focuses on finding questions with similar grammatical structure and usage. Therefore, the current retrieval and recommendation techniques will not be suitable and effective. In this paper, we propose a syntactic content-based approach which identifies grammatical structure and usage of the query and database questions for grammar question recommendation.

4. English Grammar Learning System

In this research, we developed an English Grammar Learning System, called LearnGrammar. Figure 1 shows the system architecture of LearnGrammar which consists of the following three modules: Learn, Test, and Question Recommendation. The user can learn concept description and practice grammar questions in Learn module. The user can take adaptive testing in Test module, and his historical performance will be kept in history database, which can be viewed later. For any question in history database that was taken by the user, the Question Recommendation

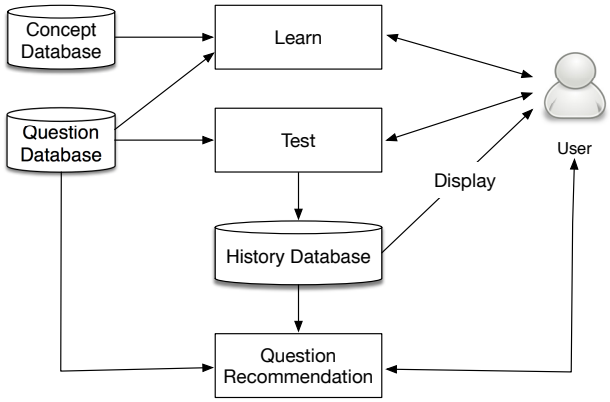

Figure 1: System Architecture of LearnGrammar

module will recommend questions with similar grammatical structure and usage to the user to help them practice and have a better understanding of the corresponding grammar knowledge. As this paper mainly focuses on discussing the proposed grammar question recommendation approach, we discuss briefly on each of the modules and functions provided in the LearnGrammar system as follows:

- Learn - It consists of two functions: Grammar Concept and Exercise. Grammar Concept provides the description of grammar concepts with examples according to grammar topics. It aims to explain the grammar concepts in details for the user to understand the concepts clearly. Exercise provides English grammar questions based on topics for the users to practice. The grammar questions are organized according to the following topics: Noun, Pronoun, Determiner, Numeral, Proposition, Adjective, Adverb, Conjunction, Clause, Verb and Tense, Modal Verb, Phrasal verb, etc.

- Test - It supports online computerized adaptive testing (CAT) Segall (2005) based on Item Response Theory (IRT). In Test module, the test questions are generated dynamically and attempted by the user online. It selects questions of suitable difficulty levels according to the user's ability, evaluates the responses, and estimates the user's ability according to the responses. Based on the predefined termination conditions, it can evaluate the user's proficiency. The advantage of this approach is to enable the estimation of the user's ability without the need of attempting all the questions as commonly needed in traditional full paper testing environment.

  Specifically, in an IRT based CAT, each test item contains some information of the test taker and can be represented in an Item Information Function (IIF). Test items are then selected by maximizing the amount of information from the IIF. The equation of Item Information Function (IIF) is shown as follow:

$$I_i(\theta_s) = P_i(\theta_s)(1 - P_i(\theta_s)) \tag{1}$$

  where $P_i(\theta_s)$ is the probability of a correct response to item i given the user ability θ_s, which is computed as:

$$P_i(\theta_s) = \frac{1}{1 + e^{-(\theta_s - b_i)}} \tag{2}$$

  where b_i is the item difficulty parameter take a range of values between -2 and +2.

  The user ability θ_s is estimated based on user's response:

$$\theta_s = \theta_{s-1} + \frac{\sum_{i=1}^{n}[u_i - P_i(\theta_{s-1})]}{\sum_{i=1}^{n} I_i(\theta_{s-1})} \tag{3}$$

  where θ_s refers to the new estimated ability level and θ_{s-1} refers to the previous estimated ability level, n is the current number of items tested, $u_i \in \{0, 1\}$ is the response detection value, u_i=1 represents the user submit correct response and u_i=0 represents the user submit an incorrect response.

  The system tracks the performance of the user in Test module into History Database and the user can view her performance results. Specifically, the system records the test scores of each test and the correctness of each attempted question.

- Question Recommendation - In Question Recommendation module, the user selects a question in her performance history, the system will recommends related grammar questions to the user. In particular, the user will be interested to know similar grammar questions for those questions that she had answered incorrectly. Once a question is selected from the list of questions attempted in a test, the related questions will be retrieved from the question database and displayed to the user. Figure 2 shows the system interface on grammar question recommendation.

5. Proposed Approach for Grammar Question Recommendation

In this section, we proposed a syntactic parse-key tree based approach for content-based grammar question recommendation. Given a query question, we aim to find similar questions in terms of grammatical structure and usage similarity from a database of questions. Figure 3 shows our proposed grammar question recommendation approach which consists of the following five processes: Parse-key Tree Construction, Parse-key Tree Similarity, Conceptual Similarity, Textual Similarity and Ranking. The Parse-key Tree Construction process constructs the parse-key trees for the query question and database questions. Then, the Parse-key Tree Similarity process computes the similarity between the parse-key trees of the

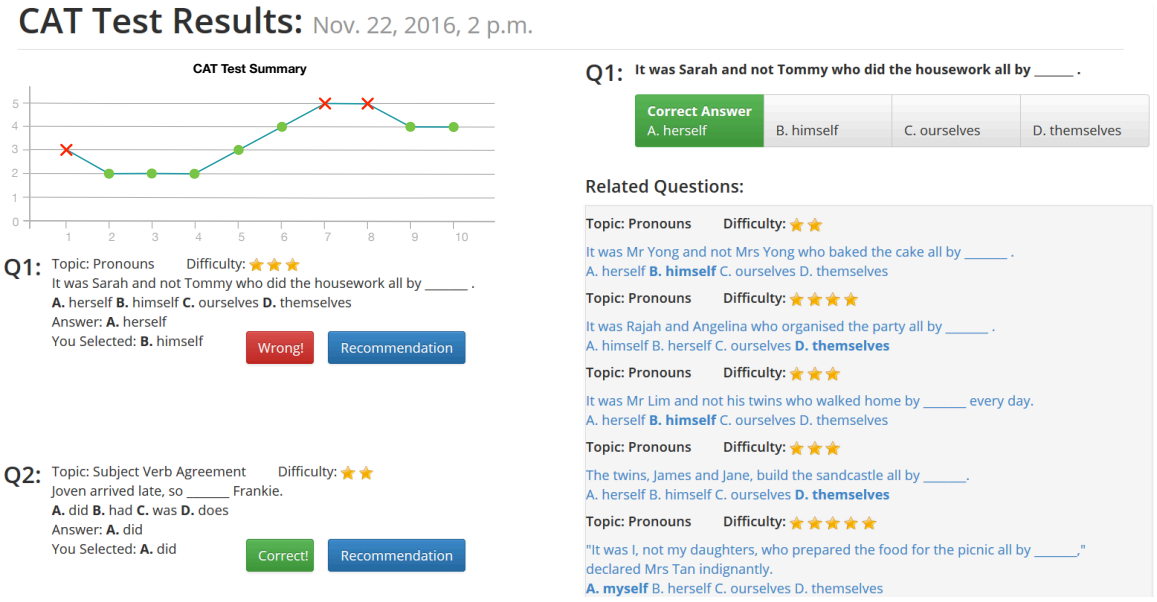

Figure 2: System Interface for Question Recommendation

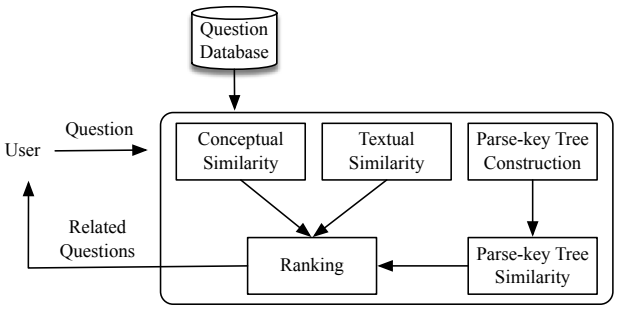

Figure 3: Proposed Question Recommendation Approach

query question and each of the database questions. The Conceptual Similarity and Textual Similarity processes compute the conceptual similarity and textual similarity between the query question and database questions. Finally, the combined similarity scores will be passed to the Ranking process for ranking.

5.1. Parse-key Tree Construction

As discussed earlier, textual content alone is not sufficient for capturing the grammatical usage of English grammar questions. In this paper, we propose a parse-key tree which is based on syntactic parse tree to effectively capture the grammatical usage of a question according to the blank or answer position. It consists of three steps: word-blank distance extraction, syntactic parsing and tree construction.

5.1.1. Word-blank Distance Extraction

The position of the blank space in the question stem may provide hint on the grammatical focus of the query question. For example, the sample query question "The lady whom you saw performing on stage is our favorite English teacher." has the focus on clause, while the database question "The lady whom you saw performing on stage is our favorite English teacher." has the focus on preposition.

To identify the grammatical focus of a query question or database question, we define word-blank distance to capture the relative position of each word to the blank position in the query question or database question.

Definition (Word-blank Distance). *A grammar question stem is represented as a sequence of elements, where each element is either a word, punctuate, or the blank space. For a word w, its position $p(w)$ is defined as the index of w in the sequence. Let w_b be the blank space in the question stem. The word-blank distance $D(w)$ of a word w in the question is defined as the distance between the word position $p(w)$ and the position of blank space $p(w_b)$, i.e.,*

$$D(w) = p(w) - p(w_b)$$

Example Consider the question stem "lady ___ you saw". The position of blank space is 2, since it is the second element in the stem. Similarly, the position of the word "lady" is 1. Thus, the word-blank distance of "lady" is $D(''lady") = 1 - 2 = -1$.

5.1.2. Syntactic Parsing

This step is used to capture the syntactic information from a grammar question through the parse tree obtained by using the Stanford parser Chen and Manning (2014) to parse the question. A parse tree is an ordered, rooted syntactic tree that represents the grammatical structure of a sentence. Before parsing the grammar question, we first reconstruct the grammar question by inserting the correct answer into the blank position of the question stem. For example, the reconstructed sample query question with the

Table 2: Some Abbreviations used in the Parse Tree

Abbreviations	Meaning
S	Sentence
NP	Noun phrase
VP	Verb phrase
SBAR	Subordinate clause
NN	Noun, singular or mass
WHNP	Wh-noun phrase
WP	Wh-pronoun
PRP	Personal pronoun
VBD	Verb, past tense

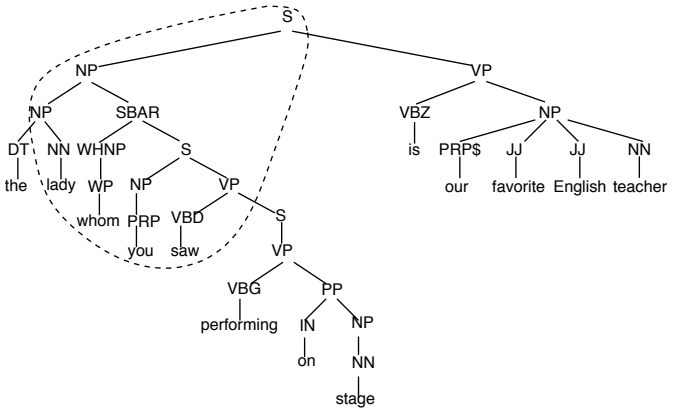

Figure 4: Parse Tree for the Sample Query Question

answer *"whom"* is *"The lady <u>whom</u> you saw performing on stage is our favorite English teacher."*.

Fig. 4 shows the parse tree of the sample query question. Some abbreviations used in Figure 4 are shown in Table 2. The leaf nodes in Figure 4 are the words in the sample query question. The leaf nodes' parents (i.e. pre-terminal nodes) are their part-of-speech (POS) tags, e.g. noun, verb, pronoun, etc.

5.1.3. Tree Construction

The syntactic parse tree can be used to represent the grammatical structure of a grammar question. However, based on our empirical experiments, it is difficult to find the grammatical focus of an English grammar question if we use the entire parse tree. Here, we incorporate the word-blank distance into the parse tree, and propose *parse-key tree* as a sub-parse tree to represent the grammatical focus of a grammar question.

Before discussing parse-key tree, we first give the definition on neighbor nodes as follows:

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 parse tree of the query question stem *"The lady <u>whom</u> you saw performing on stage is our favorite English teacher."* shown in Figure 4. With the parameters $M = -1$ and $N = 2$, we can extract the neighbor nodes {"lady", "whom", "you". "saw"}.

Algorithm 1: Parse-key Tree Construction

Input: Q_{stem}: A question stem; A: Correct Answer
Output: T_{PK}: A parse-key tree
1 Compute word-blank distances for the words in Q_{stem}
2 $Q \leftarrow$ sentence constructed by Q_{stem} and A
3 $T \leftarrow$ parse tree obtained from Q
4 *neighbor* $\leftarrow$ neighbor nodes extracted from T
5 *predecessor* $\leftarrow$ predecessor nodes of neighbor nodes extracted from T
6 $T_{PK} \leftarrow$ subtree consisting of nodes in *neighbor* and *predecessor*
7 **for** each leaf node n in T_{PK} **do**
8 | Add sibling node labeled with n's word-blank distance to n
9 **return** T_{PK}

Definition (Parse-key tree). *Given a grammar question with a blank position, 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 in a grammar question stem Q_{stem} as input and returns the parse-key tree T_{PK}. First, it computes the word-blank distance for the words in Q_{stem} (line 1). Then, it filled the correct answer into the question stem to construct the a new sentence (line 2). Next, it uses the Stanford parser to parse the new sentence to obtain the parse tree (line 3). Then, it constructs a parse-key tree from the parse tree and word-blank distances (lines 4-8). 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 shape in Figure 4 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, where word-blank distance of the word at blank position is "0". Finally, the parse-key tree T_{PK} is returned. The parse-key tree of the sample query question is shown in Figure 5.

5.2. Parse-key Tree Similarity

In the Parse-key Tree Construction process, we construct the parse-key trees of the query question and each database question, and compute the similarity between them. The proposed parse-key tree similarity process consists of two steps: calculating structural similarity and calculating POS order similarity.

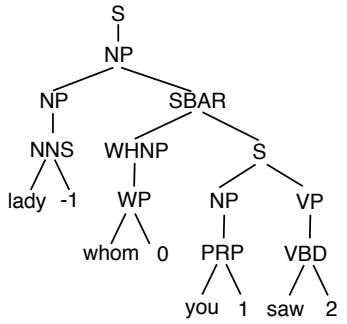

Figure 5: Parse-key Tree for the Sample Query Question with $M = -1$ and $N = 2$

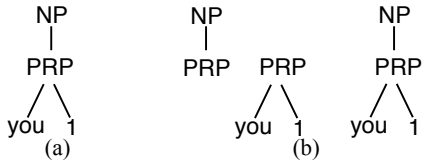

Figure 6: An Example for Tree Fragment

5.2.1. Calculating Structural Similarity

This step aims to capture the structural similarity between parse-key trees of the query question and each database question. Tree kernel Collins and Duffy (2001) can be used to capture the structural information from a syntactic tree Tymoshenko and Moschitti (2015); Wang et al. (2009); Bloehdorn and Moschitti (2007); Moschitti (2006). However, tree kernel does not fit well into parse-key tree. Therefore, we propose a new method for structural similarity calculation.

The main idea behind structural similarity is to calculate the common tree fragments between two parse-key trees. Before introducing tree fragments, we first give the definition on production.

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.*

For example, consider the node "PRP' in Figure 7. The children of "PRP" are "you" and "1", so its production is "PRP $\rightarrow$ you 1".

The definition of tree fragment is given as follows:

Definition (Tree Fragment). *Given a parse-key tree, a tree fragment is a sub-graph which has more than one node and must be the entire production at the node.*

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 entire production.

Note that the punctuation nodes can change the grammatical structure of a grammar question, while word-blank distance nodes may provide hint on the grammatical focus

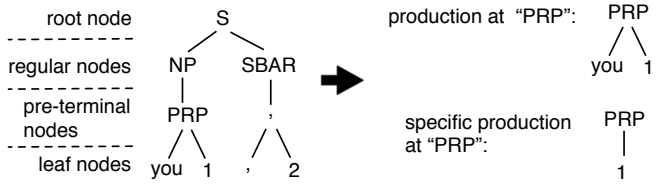

Figure 7: An Example for Production and Specific Production

of a query question. Therefore, we define specific production to enhance the effect of such nodes for the computation of common tree fragments.

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 7. The children of the pre-terminal node "PRP" are leaf nodes "you" and "1", where "1" is a word-blank distance node, so its specific production is " PRP $\rightarrow$ 1". The children of the pre-terminal node "," are leaf nodes "," and "2", where "," is a punctuation node, "2" is word-blank distance node. So it has two specific productions: (1) " , $\rightarrow$, " and (2) ", $\rightarrow$ 2".

The structural similarity between two parse-key trees is based on the number of common tree fragments. The common tree fragments is defined as follows.

Definition (Common Tree Fragments). *Given two tree fragments, they are common tree fragments if the production or any specific production at each node in the two tree fragments are the same.*

As defined earlier, 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 structural similarity. Next, we discuss how to compute the structural similarity, which is the total number of the common tree fragments between two parse-key trees.

Algorithm 2 shows the structural similarity calculation method. The inputs of the algorithm are two parse-key trees T_q and T_d, and the output of the algorithm is the structural similarity which is defined as 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 structural similarity $K_{T_{PK}}$ 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 $K_{T_{PK}}$ (lines 3–6).

Algorithm 2: Structural Similarity

Input: T_q: parse-key tree of query
T_d: parse-key tree of question in database
Output: $K_{T_{PK}}$: structural similarity

```
N_q ← all the nodes in T_q(exclude leaf nodes)
N_d ← all the nodes in T_d(exclude leaf nodes)
K_TPK = 0
for n_q ∈ N_q do
    for n_d ∈ N_d do
        K_TPK = K_TPK + CTF(n_q, n_d)
return K_TPK;
```

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 computed 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 Collins and Duffy (2001). Specifically, our proposed method can capture the common tree fragments from similar structures whose productions or any specific productions are the same.

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 take the production at n_q and n_d, together with either simply taking the non-terminal at the child, or taking any one of the common tree fragments at the child, for each child 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 effectively compute the number of common tree fragments rooted at n_q and n_d.

5.2.2. Calculating POS order similarity

To compute the POS order similarity, 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.*

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 the parse-key tree in Figure 5, the POS tags list of this parse-key tree is {"NN", "SYM", "PRP", "VBD"}. The LCS sequence between a parse-key tree with POS tags list {"NN", "SYM", "PRP", "VBD"} and another parse-key tree with POS tags list {"VBD", "JJ", "NN", "PRP"} is {"NN", "PRP"}.

Given two parse-key trees T_q and T_d, assuming LS is their LCS sequence, and their POS tags list are L_q and L_d, respectively. $P_{L_q}(s)$ 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. The order similarity of s in LS is defined as:

$$S_{order}(s) = \frac{1}{|P_{L_q}(s) - P_{L_d}(s)| + 1} \tag{4}$$

where $|P_{L_q}(s) - P_{L_d}(s)|$ is the position distance of s between L_q and 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 similarity POS_{order} between T_q and T_d is calculated as the summation of order similarity of all elements in their LCS sequence divided by the longer length of L_q and L_d. Given two parse-key trees T_q and T_d, their POS order similarity is defined as:

$$POS_{order}(T_q, T_d) = \frac{1}{max(|L_q|, |L_d|)} \sum_{s \in LS} \frac{1}{|P_{L_q}(s) - P_{L_d}(s)| + 1} \tag{5}$$

where $max(|L_q|, |L_d|)$ is the longer length of lists L_q and L_d.

5.2.3. Parse-key tree similarity

In order to compute the similarity score between the parse-key trees T_q and T_d, we normalize the range of the structural similarity and POS order similarity to [0, 1]. The normalized structural similarity score is defined as:

$$K'_{T_{PK}}(T_q, T_d) = \frac{K_{T_{PK}}(T_q, T_d) - K_{min}}{K_{max} - K_{min}} \tag{6}$$

where K_{max} and K_{min} are the respective maximum and minimum values from the structural similarity between the query question and each of the database questions.

The normalized POS order similarity is defined as:

$$POS'_{order}(T_q, T_d) = \frac{POS_{order}(T_q, T_d) - POS_{min}}{POS_{max} - POS_{min}} \tag{7}$$

where POS_{max} and POS_{min} are the respective maximum and minimum values from the POS order similarity be-

Table 3: Example Concepts and Words

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

Table 4: Conceptual Features for the Sample Query and Questions in Table 1

	Conceptual Features
Query	{"relative-pronoun", "clause", "question-word"}
Q1	{"relative-pronoun", "clause", "question-word"}
Q2	{"comparison-of-adjectives "}

tween the query question and each of the database questions.

The parse-key tree similarity is the combined score of structural similarity and POS order similarity. Given the parse-key trees T_q of query question Q_q and T_d of database question Q_d. The parse-key tree similarity between Q_q and Q_d is defined as:

$$S_{T_{PK}}(T_q, T_d) = k_1 K'_{T_{PK}}(T_q, T_d) + k_2 POS'_{order}(T_q, T_d) \quad (8)$$

where k_1 and k_2 are weighting factors.

5.3. Conceptual Similarity

In English grammar questions, there are some words that always have the same grammatical properties. For example, words such as "better", "best" and "worst" have the grammatical properties of "comparison-of-adjective". Therefore, we build a list of grammatical concepts based on common grammar properties. Each concept contains a set of words to represent a grammatical category. In total, there are 76 such concepts created manually. Table 3 gives some examples of conceptual categories and part of its words list. Given an English grammar question, the conceptual feature extraction process will determine whether the words at the blank position of question stem are in the conceptual words list or not. If yes, it extracts the corresponding concept name as the feature. The conceptual feature will be $\{\varnothing\}$ if the words at the blank position are not found in the conceptual words list. For the sample query question, the answer *C: whom* is in the conceptual words list of "relative-pronoun", "clause" and "question-word". Therefore, the query question's conceptual features are {"relative-pronoun", "clause", "question-word"}. Table 4 shows the conceptual features of the query question and the database questions given in Table 1. As shown in Table 4, the conceptual features of the query question are more similar to Q1 than Q2.

The similarity score for conceptual features is calculated as:

$$S_c(Q_q, Q_d) = \frac{C_{Q_q} \cap C_{Q_d}}{C_{Q_q} \cup C_{Q_d}} \quad (9)$$

Table 5: Textual Features for the Sample Query and Questions in Table 1

Question	Textual Features
Query	{whom}
Q1	{whom}
Q2	{best}

where C_{Q_q} and C_{Q_d} are conceptual features of Q_q and Q_d. If C_{Q_q} and C_{Q_d} are both empty, we set the value of $S_c(Q_q, Q_d)$ to 1.

5.4. Textual Similarity

Apart from conceptual features, we also extract textual features for the query question and the database questions. However, we focus only on those words which occur at the blank position of the question stem and extract them as textual features. In particular, we extract the answer of the query question and the database questions as the textual features. Table 5 shows the textual features of the sample query and database questions. As shown in Table 5, the textual feature for the query question is {"whom"}, and the textual features for Q1 and Q2 are {"whom"} and {"best"} respectively.

The similarity score for textual features is calculated as:

$$S_t(Q_q, Q_d) = \frac{E_{Q_q} \cap E_{Q_d}}{E_{Q_q} \cup E_{Q_d}} \quad (10)$$

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

5.5. Ranking

The grammatical similarity between the query question Q_q and each database question Q_d is calculated based on parse-key similarity, conceptual similarity and textual similarity. In the experiments which follow, we combined the sub-similarities with the regression techniques of linear ridge regression, random forests, linear kernel-based SVM-regression and RBF kernel-based SVM-regression. RBF kernel-based SVM-regression leading to the best result (Table 8) . After computing the similarity scores between the query question and all the database questions, we rank the database questions according to their similarity scores.

6. Performance Evaluation

In this section, we discuss the performance evaluation of the proposed approach including experimental setup, evaluation metrics, parameter tuning and performance results.

6.1. Experimental Setup

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

The relevance between each query and database question pair has a relevance label manually annotated on a 5-level relevance scale(bad, fair, good, excellent, and perfect), where bad represent the database question is irrelevant to the query and perfect represent the database question is the most relevant to the query. We split the dataset into two sets: training and testing. The split was done randomly taking care that the topic distribution in both the sets remained proportional.

6.2. Evaluation Metrics

In the experiments, we use the following three metrics for evaluation:

- Precision at rank K (P@K): It measures the percentage of relevant questions ranked at top K position. High precision represent the algorithm return substantially more relevant results than irrelevant ones. The equation of P@K 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 [1] at rank K (MAP@K): It measures the mean of the average precision scores for each query. It considers whether the relevant items tend to get ranked highly. The equation of MAP@K 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 [2] (MRR): It measures the average of the reciprocal ranks of results for a sample of queries. MRR evaluate whether the highest-ranked relevant question get ranked highly. The equation of MRR 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 question for the query q.

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

6.3. Parameter Tuning and Learning

We varied the value of M and N to construct the parse-key tree and compute the parse-key tree similarity. Then we used regressions with 3-fold cross-validation to model the similarity score. Specifically, the value of M is taken from $[-5, 0]$, while the value of N is from $[0, 5]$. We used linear ridge-regression, SVM-regressions (linear kernel and RBF kernel) and Random Forests to build the models. For ridge regression, we selected the optimal ridge coefficient by varying it from 1 to 1000. For SVMs(linear-kernel and RBF-kernel), we varied the penalty factor $C \in \{2^{-5}, 2^{-3}, \ldots, 2^{13}, 2^{15}\}$, single parameter $\gamma \in \{2^{-15}, 2^{-13}, \ldots, 2^{1}, 2^{3}\}$ and epsilon-tube $\epsilon \in \{2^{-15}, 2^{-13}, \ldots, 2^{1}, 2^{3}\}$. For random forest, we varied the number of estimators from 2 to 100.

We set the parameters $M = -1$ and $N = 2$ according to the cross-validation results. In addition, since RBF-kernel SVM outperforms the other ranking methods in the cross-validation results, we use RBF-kernel SVM as the ranking model in our system.

6.4. State-of-the-art Methods Used for Performance Comparison

In the experiments, apart from evaluating our proposed approach, we also compare our proposed approach with the following classical text and sentence retrieval methods:

- BM25: This is the text retrieval method, which is based on statistical analysis of term occurrences in the text.
- PhRank Maxwell and Croft (2013): This is the retrieval model based on the term ranking algorithm, PhRank.
- C-DSSM Shen et al. (2014): This is a state-of-the-art deep matching model for Web search. This method incorporates a convolutional-pooling structure over word sequences to learn low-dimensional representations for search queries and Web documents.
- Tree Kernel (TK) Collins and Duffy (2001): This is the method based on syntactic parse tree and the original tree kernel function for counting common subtrees between syntactic trees.

[1] https://www.kaggle.com/wiki/MeanAveragePrecision
[2] https://en.wikipedia.org/wiki/Mean_reciprocal_rank

Table 6: Performance Results for the Proposed Approach

	P@1	P@3	P@5	MAP@3	MAP@5	MRR
Structural	87.5	69.4	61.7	88.5	86.0	91.3
POS order	66.7	63.9	53.3	72.2	72.1	74.1
Parse-key tree	87.5	76.4	69.2	89.6	88.1	91.6
Parse-key tree + conceptual	91.7	83.3	74.2	93.1	92.0	95.1
Parse-key tree + conceptual + textual	100	93.1	86.7	99.3	98.0	100

Table 7: Performance Comparison with Other Methods

	P@1	P@3	P@5	MAP@3	MAP@5	MRR
BM25	26.7	23.3	22.0	38.6	42.1	44.5
PhRank	31.3	20.8	13.8	37.0	37.3	48.6
C-DSSM	29.2	19.4	18.3	30.6	31.3	39.0
TK	33.3	23.6	22.5	43.1	42.4	46.9
SAM	16.7	27.8	25.8	35.4	38.4	41.0
Our approach	100	93.1	86.7	99.3	98.0	100

- Syntactic Analysis Method (SAM)Carmel et al. (2014): This method considers both statistical features and syntactic features.

6.5. Performance Results

In the experiment, we firstly evaluate the performance of different combinations of the proposed similarity methods described in Section 5. We then compare the performance of the proposed approach with other classical text and sentence retrieval methods.

Table 6 shows the performance results of different combinations of the proposed similarity methods, where structural, POS order, parse-key tree, conceptual and textual correspond to structural similarity, POS order similarity, parse-key tree similarity, conceptual similarity and textual similarity respectively as discussed in Section 5. We can observe that the performance of parse-key tree similarity (i.e. the combination of structural similarity and POS order similarity) in the proposed approach is better than that of the individual structural similarity or POS order similarity. In addition, the performance of using parse-key tree similarity is also better than the individual conceptual similarity and textual similarity. Combining parse-key tree similarity and conceptual similarity is able to improve the performance by up to 8% when compared with using only the parse-key tree similarity. It shows that the conceptual similarity is useful for improving the retrieval performance. Finally, the combined similarity measures in the proposed approach have achieved the best overall performance result. It shows that the proposed similarity measures are well complementary to each other in our proposed approach.

We also compare our proposed approach with other classical text and sentence retrieval methods. Table 7 shows the performance results of the comparison. We can observe that for all metrics, our proposed approach has performed significantly better than all other comparison methods. Generally, our proposed approach has improved by more than 50% for all metrics when compared to methods from BM25, PhRank, C-DSSM, TK and SAM. It shows that our proposed approach is more effective than other methods in the task of grammar question retrieval and recommendation.

As shown in the performance results, our approach has achieved better performance than other methods because the proposed parse-key tree can effectively capture the grammatical structure and usage of the questions and able to rank questions based on both structural similarity and POS order similarity. In addition, conceptual similarity and textual similarity which capture the conceptual and textual information of grammar questions can also help in improving the performance.

Table 8 shows the performance results of different models. We observe that SVM-RBF lead to the best results.

7. Conclusion and Future Work

In this paper, we have proposed a content-based approach for English grammar question recommendation. In the proposed approach, a syntactic tree structure, called parse-key tree, is used to capture the grammatical structure of grammar questions. The proposed approach computes the parse-key tree similarity, conceptual similarity and textual similarity of the question query and database questions for grammar question recommendation. The proposed grammar question recommendation approach has been incorporated into a Web-based English grammar learning system. Based on the proposed question recommendation approach, the system can recommend relevant grammar questions according to users' preferences and needs. The performance results have shown that our proposed approach has significantly outperformed other classical text and sentence retrieval methods on grammar question recommendation.

Table 8: Performance Performance across models - All similarities

	P@1	P@3	P@5	MAP@3	MAP@5	MRR
Ridge	100	91.7	85.8	99.3	97.5	100
Random Forest	79.2	77.8	74.2	86.5	87.8	87.6
SVM - RBF	100	93.1	86.7	99.3	98.0	100
SVM - Linear	100	91.7	85.8	99.3	97.5	100

One application of the proposed recommendation method in real-world is to help users learning English in massive open online course (MOOC) and e-Learning system. Additionally, through changing the parsing method, the proposed parse-key similarity can be applied to other languages like Chinese.

The recommendation system in this work only focuses on the content similarity between English grammar questions. In future research, we are planning to extend the system through social interactions by using the information among groups of learners. Additionally, jointly using neural network methods, e.g. deep learning, fuzzy SVM Wang et al. (2017b), with our approach is an interesting and promising research direction.

References

Barr, C., Jones, R., Regelson, M., 2008. The linguistic structure of english web-search queries. In: Proceedings of the conference on empirical methods in natural language processing. Association for Computational Linguistics, pp. 1021–1030.

Barragáns-Martínez, A. B., Costa-Montenegro, E., Burguillo, J. C., Rey-López, M., Mikic-Fonte, F. A., Peleteiro, A., 2010. A hybrid content-based and item-based collaborative filtering approach to recommend tv programs enhanced with singular value decomposition. Information Sciences 180 (22), 4290–4311.

Bloehdorn, S., Moschitti, A., 2007. Structure and semantics for expressive text kernels. In: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. ACM, pp. 861–864.

Carmel, D., Mejer, A., Pinter, Y., Szpektor, I., 2014. Improving term weighting for community question answering search using syntactic analysis. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, pp. 351–360.

Chen, C.-M., Chung, C.-J., 2008. Personalized mobile english vocabulary learning system based on item response theory and learning memory cycle. Computers & Education 51 (2), 624–645.

Chen, D., Manning, C. D., 2014. A fast and accurate dependency parser using neural networks. In: EMNLP. pp. 740–750.

Chinkina, M., Kannan, M., Meurers, D., 2016. Online information retrieval for language learning. ACL 2016, 7.

Collins, M., Duffy, N., 2001. Convolution kernels for natural language. In: Advances in neural information processing systems. pp. 625–632.

Damiano, R., Gena, C., Lombardo, V., 2015. Leveraging social semantic components in executable environments for learning. Expert Systems 32 (2), 277–292.

Ding, X., Liu, T., Duan, J., Nie, J.-Y., 2015. Mining user consumption intention from social media using domain adaptive convolutional neural network. In: AAAI. Vol. 15. pp. 2389–2395.

Dwivedi, P., Bharadwaj, K. K., 2015. e-learning recommender system for a group of learners based on the unified learner profile approach. Expert Systems 32 (2), 264–276.

Higashinaka, R., Kobayashi, N., Hirano, T., Miyazaki, C., Meguro, T., Makino, T., Matsuo, Y., 2016. 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. Springer, pp. 15–26.

Hsu, C.-K., Hwang, G.-J., Chang, C.-K., 2010. Development of a reading material recommendation system based on a knowledge engineering approach. Computers & Education 55 (1), 76–83.

Hsu, C.-K., Hwang, G.-J., Chang, C.-K., 2013. A personalized recommendation-based mobile learning approach to improving the reading performance of efl students. Computers & Education 63, 327–336.

Hsu, M.-H., 2008. A personalized english learning recommender system for esl students. Expert Systems with Applications 34 (1), 683–688.

Huang, W., Wu, Z., Chen, L., Mitra, P., Giles, C. L., 2015. A neural probabilistic model for context based citation recommendation. In: AAAI. pp. 2404–2410.

Huang, Y.-M., Huang, Y.-M., Huang, S.-H., Lin, Y.-T., 2012. A ubiquitous english vocabulary learning system: Evidence of active/passive attitudes vs. usefulness/ease-of-use. Computers & Education 58 (1), 273–282.

Lease, M., 2007. Natural language processing for information retrieval: the time is ripe (again). In: Proceedings of the ACM first Ph. D. workshop in CIKM. ACM, pp. 1–8.

Li, J. W., Chang, Y. C., Chu, C. P., Tsai, C. C., 2012. A self-adjusting e-course generation process for personalized learning. Expert Systems with Applications 39 (3), 3223–3232.

Liu, Q., Chen, E., Xiong, H., Ding, C. H., Chen, J., 2012. Enhancing collaborative filtering by user interest expansion via personalized ranking. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (1), 218–233.

Lo, J.-J., Yeh, S.-W., Sung, C.-S., 2013. Learning paragraph structure with online annotations: An interactive approach to enhancing efl reading comprehension. System 41 (2), 413–427.

Lops, P., De Gemmis, M., Semeraro, G., 2011. Content-based recommender systems: State of the art and trends. In: Recommender systems handbook. Springer, pp. 73–105.

Maxwell, K. T., Croft, W. B., 2013. Compact query term selection using topically related text. In: Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval. pp. 583–592.

Moschitti, A., 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In: European Conference on Machine Learning. Springer, pp. 318–329.

Park, J. H., Croft, W. B., Smith, D. A., 2011. A quasi-synchronous dependence model for information retrieval. In: Proceedings of the 20th ACM international conference on Information and knowledge management. ACM, pp. 17–26.

Qian, X., Feng, H., Zhao, G., Mei, T., 2014. Personalized recommendation combining user interest and social circle. IEEE transactions on knowledge and data engineering 26 (7), 1763–1777.

Robertson, S., Zaragoza, H., 2009. The probabilistic relevance framework: BM25 and beyond. Now Publishers Inc.

Robertson, S. E., Walker, S., Beaulieu, M., Gatford, M., Payne, A., 1996. Okapi at trec-4. NIST SPECIAL PUBLICATION SP, 73–96.

Santos, O. C., Boticario, J. G., 2015. User-centred design and educational data mining support during the recommendations elicitation process in social online learning environments. Expert Systems 32 (2), 293–311.

Schubotz, M., Grigorev, A., Leich, M., Cohl, H. S., Meuschke, N., Gipp, B., Youssef, A. S., Markl, V., 2016. Semantification of identifiers in mathematics for better math information retrieval. energy 2 (5), 10.

Segall, D. O., 2005. Computerized adaptive testing. Encyclopedia of social measurement 1, 429–438.

Seo, Y.-D., Kim, Y.-G., Lee, E., Baik, D.-K., 2017. Personalized recommender system based on friendship strength in social network services. Expert Systems with Applications 69, 135–148.

Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G., 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, pp. 101–110.

Song, Y.-I., Han, K.-S., Kim, S.-B., Park, S.-Y., Rim, H.-C., 2008. A novel retrieval approach reflecting variability of syntactic phrase representation. Journal of Intelligent Information Systems 31 (3), 265–286.

Thorat, P. B., Goudar, R., Barve, S., 2015. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. International Journal of Computer Applications 110 (4).

Tymoshenko, K., Moschitti, A., 2015. Assessing the impact of syntactic and semantic structures for answer passages reranking. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. ACM, pp. 1451–1460.

Wang, K., Ming, Z., Chua, T.-S., 2009. 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. ACM, pp. 187–194.

Wang, S., Li, P., Chen, P., Phillips, P., Liu, G., Du, S., Zhang, Y., 2017a. Pathological brain detection via wavelet packet tsallis entropy and real-coded biogeography-based optimization. Fundamenta Informaticae 151 (1-4), 275–291.

Wang, S., Li, Y., Shao, Y., Cattani, C., Zhang, Y., Du, S., 2017b. Detection of dendritic spines using wavelet packet entropy and fuzzy support vector machine. CNS & Neurological Disorders-Drug Targets (Formerly Current Drug Targets-CNS & Neurological Disorders) 16 (2), 116–121.

Wang, S., Lu, S., Dong, Z., Yang, J., Yang, M., Zhang, Y., 2016a. Dual-tree complex wavelet transform and twin support vector machine for pathological brain detection. Applied Sciences 6 (6), 169.

Wang, Y., Wu, S., Li, D., Mehrabi, S., Liu, H., 2016b. A part-of-speech term weighting scheme for biomedical information retrieval. Journal of Biomedical Informatics.

Zhang, W., Ming, Z., Zhang, Y., Nie, L., Liu, T., Chua, T.-S., 2012. The use of dependency relation graph to enhance the term weighting in question retrieval. In: COLING. pp. 3105–3120.

Zhang, Y., Wang, S., Dong, Z., 2014a. Classification of alzheimer disease based on structural magnetic resonance imaging by kernel support vector machine decision tree. Progress In Electromagnetics Research 144, 171–184.

Zhang, Y., Wang, S., Ji, G., Phillips, P., 2014b. Fruit classification using computer vision and feedforward neural network. Journal of Food Engineering 143, 167–177.

Zhang, Y., Wang, S., Phillips, P., Ji, G., 2014c. Binary pso with mutation operator for feature selection using decision tree applied to spam detection. Knowledge-Based Systems 64, 22–31.