An End-to-end Tag-based Recommendation System for Verbal Reasoning Questions

ZHIXIONG YUE, Southern University of Science and Technology

ABSTRACT

Developing a verbal reasoning question[1] recommendation system is a good way to help the GRE® test takers improve their verbal reasoning abilities by practicing questions more efficiently. As there are a great number of GRE® verbal reasoning practice questions and limited practicing time for test takers, it is impossible to practice all kinds of questions at the same time. We believe that the personalized recommendation system should be built up according to the special features of a certain examinee, and forming professional recommendation systems for different questions. In this paper, based on the examinee's current verbal reasoning ability obtained from the historical practicing accuracy and difficulty, we propose an End-to-end Tag-based Recommendation System (ETRS) for task takers to optimize practice effect. Code of this paper can be found on *https://github.com/Oliver-Q/ETRS-for-Verbal-Reasoning-Questions*. Experimental results show the system is feasible and effective.

KEYWORDS

Recommender system, Personalization service, User tagging, Text tagging, Cold-start problem, Nature language processing

1 INTRODUCTION

According to A Snapshot of the Individuals Who Took the GRE® General Test, a total of 584,677 examinees took the GRE® General Test between July 1, 2015, and June 30, 2016. The verbal reasoning abilities are huge uneven for different task taker. Since everyone needs to do exercise before they go to take a real exam, one simple recommendation system cannot provide a personalized service for everyone.

Verbal reasoning questions appear in several formats, Text Completion question is what we focused on and discussed in detail below. As for verbal reasoning questions, about half of the measure requires you to read passages and answer questions on those passages. The other half requires you to read, interpret, and complete existing sentences, groups of sentences, or paragraphs. Many, but not all, of the questions are standard multiple-choice questions, in which you are required to select a single correct answer; others ask you to select multiple correct answers; and still others ask you to select a sentence from the passage. The number of choices varies depending on the type of question.

1.1 Text Completion Question

As mentioned above, skilled readers do not simply absorb the information presented on the page; instead, they maintain a constant attitude of interpretation and evaluation, reasoning from what they have read so far to create a picture of the whole and revising that picture as they go. Text Completion questions test this ability by omitting crucial words from short passages and asking the test taker to

XX

[1] Text Completion questions in GRE® revised General Test. The Verbal Reasoning measure assesses your ability to analyze and evaluate written material and synthesize information obtained from it, to analyze relationships among component parts of sentences, and to recognize relationships among words and concepts.

[2] The sample question set is fetched from http://gre.kmf.com/practisenew/tc/3/54

use the remaining information in the passage as a basis for selecting words or short phrases to fill the blanks and create a coherent, meaningful whole. [1]

See a sample of text completion question in Figure1.1. The question showed below is composed of three sentences and has three blanks. Three answer choices per blank function independently; which is to say, selecting one answer choice for one blank does not affect what answer choices you can select for another blank. Single correct answer, consisting of one choice for each blank; no credit for partially correct answers.

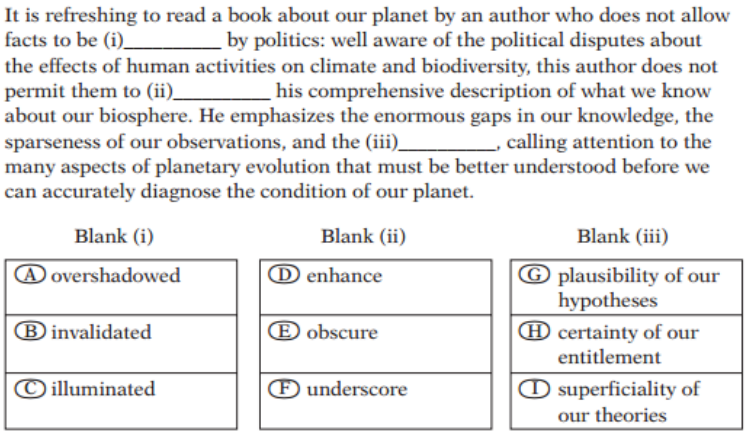

Figure 1.1 A Sample Text Completion Question of GRE® General Test

Conventionally, your GRE Verbal Reasoning skills can be sharpened by working your way through these question sets. Begin with the easy sets and then move on to the medium-difficulty and hard sets. Review the answer explanations carefully, paying particular attention to the explanations for questions that you answered incorrectly. Traditional training way which guides the test taker to exercise the questions in a specific or random order is becoming less effective because it is lack of personalization.

As years past since the birth of GRE® General Test, the practicing question sets are becoming increasingly larger. Practice questions are infinite while the examinee's time is limited. Depending on the sort of question text, various personalized recommender systems can be built up to guide the examinees in a large question feature space. For such frequently-examining knowledge points as clauses, adversatives, and pronouns, recommendation systems can be developed to compensate his blind spots by analyzing his historical fallible difficulty.

1.2 End-to-end Tag-based Recommendation System(ETRS)

In this approach, we present a tag-based recommendation system for those examinees, especially for who don't know his missing knowledge points. When a test taker is going practicing, some distinctive features of questions may get him into some trouble. First, compared his verbal reasoning abilities with other practicer, the historical practicing accuracy is objective and precise. As the new practice question comes out, we managed to add knowledge points tags for the question via nature language processing tools. Second, accompanied by newly-practiced questions, a great number of personal fallible difficulties come forth to personalize the user tags. Even though knows the details of the missing knowledge points, he still does not know whether he has compensated his blind spots. Last, we recommend relative questions based on the common tags between user and questions until the user no longer makes mistake of the same knowledge points.

The proposed system aims to assist an examinee to navigate the questions feature space in an interactive way in which the examinee has his own fallible difficult in each feature dimension so that the examinee can find the optimal question to compensate his blind spots. We have also built up a system of this kind for GRE® verbal reasoning practice questions recommendations. The experimental results show that both systems can give sensible recommendations, and adapt to examinees' up-to-date knowledge points. For user who may not have many historical practicing, the ETRS will manage to know the user's tag first. In this situation, practicing accuracy can work as a guidance for new users.

The remainder of the paper is organized as follows. In Section 2, research background is expatiated, including nature language processing, cold-start problem and recommendation system. Section 3 gives detail information and algorism of the recommendation system. Section 4 reports the experimental process and the results of the study. Finally, the conclusion and future work are given in Section 5.

2 RESEARCH BACKGROUND

The purpose of this research is to build up an end-to-end recommendation system based on the common tags between questions and users. In order to add tags for questions automatically, we introduced some nature language processing approach. To optimize the recommendation effect for both new questions and new users, we clever used the accuracy matching mechanism to solve the cold-start problem in our system. At last, we showed our tag-based feature compare to traditional recommend strategy.

2.1 Nature Language Processing(NLP)

Text completion question of verbal reasoning is nothing more but nature language of English speakers. Not to mention the question's a high degree of correspondence to the reality. Syntax and structure usually go hand in hand, where a set of specific rules, conventions, and principles usually govern the way words are combined into phrases, phrases get combines into clauses, and clauses get combined into sentences. Words are the smallest units in a language that are independent and have a meaning of their own. Some significant words with logical meaning represent the solution points for different questions. These words are predictable and easy to extract from the context of the questions. We used nature language processing tools to analyze the question and obtain the keywords for tagging automatically.

2.2 Cold-start problem

Without a large amount of user data and items features, it hard to allow the users to be satisfied with the recommendation results and willing to use the recommendation system. The cold start problem mainly divided into three categories: new items, new users and new system. For new released questions, we automatically extract the tags with nature language processing tools and statistic the overall exercise accuracy. For new started users, we recommend the question matching their historical accuracy even though only several questions have been done. In this way, the whole system can quickly get data to perform better and have a fair recommend effect at the very beginning.

2.3 Tag-based Recommendation

A recommendation system can provide personalized information services in different ways; it depends on whether the system has been recording and analyzing a user's previous preferences. In item-based recommendation system, approach looks into the set of items the target user has rated and computes how similar they are to the target item i and then selects k most similar items $\{i_1, i_2, \ldots, i_k\}$. At the same time their corresponding similarities $\{s_{i1}, s_{i2}, \ldots, s_{ik}\}$ are also computed. Once the most similar items are found, the prediction is then computed by taking a weighted average of the target user's ratings on these similar items. [2]

Unlike the above system which needs lots of users' and items' data and quantities of calculations requiring for a powerful machine. Tag-based recommendation system of this type aim to assist an examinee to find out what is his really shortage, when he can simply practice the type of questions and detect his fallible difficulty.

3 EXPERIMENTAL AND COMPUTATIONAL DETAILS

3.1 Text Tokenization and Question Tagging

Text analytics, also known as text mining, is the methodology and process followed to derive quality and actionable information and insights from textual data. This involves using NLP, information retrieval, and machine learning techniques to parse unstructured text data into more structured forms and deriving patterns and insights from this data that would be helpful for the end user.

Natural Language Toolkit(NLTK) is a leading platform for building Python programs to work with human language data, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum [3]. Shallow parsing, also known as light parsing or chunking, is a technique of analyzing the structure of a sentence to break it down into its smallest constituents (which are tokens such as words) and group them together into higher-level phrases. In shallow parsing, there is more focus on identifying these phrases or chunks rather than diving into further details of the internal syntax and relations inside each chunk, like we see in grammar-based parse trees obtained from deep parsing. The main objective of shallow parsing is to obtain semantically meaningful phrases and observe relations among them. Figure 3.1.1 is the preceding output is the raw shallow-parsed sentence tree for our sample question. We leveraged the pattern package here to create a shallow parser to extract meaningful chunks out of sentences.

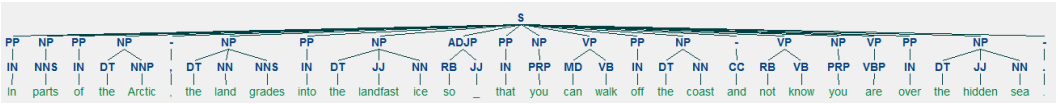

Figure 3.1.1 Visual representation of a shallow parsed tree for a sample question text.

We explored some techniques by which we can build our own POS taggers and will be leveraging some classes provided by NLTK for doing so. In Figure 3.1.2, we can see there are several verbal texts representing questions which can be assigned to various categories of adversative, clauses and so on. Initially, these questions are all present together, just as a text corpus has various texts in it. Once it goes through a text classification system, represented as a grey box here, we can see that each document is assigned to one specific class or category we had defined previously. Thus, we can achieve content-based tags of questions automatically.

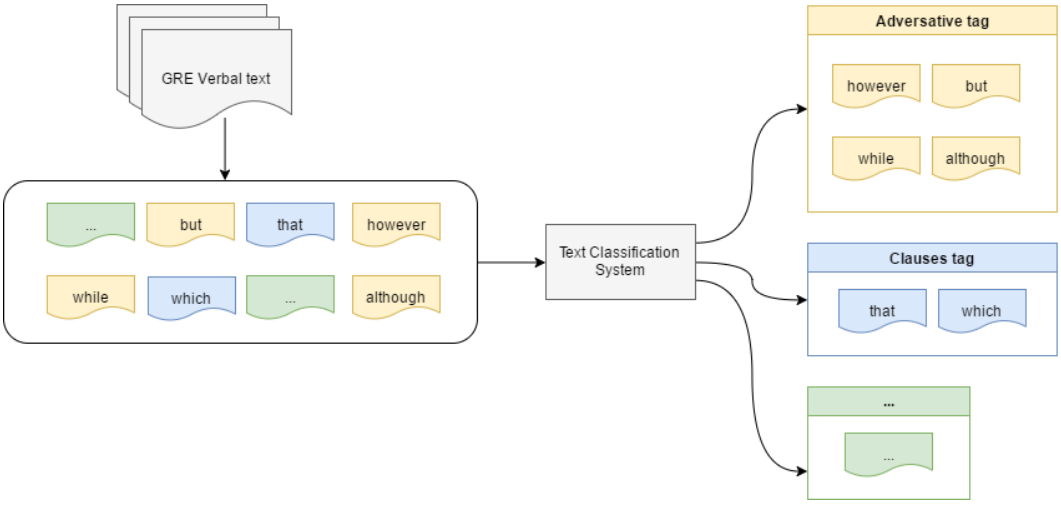

Figure 3.1.2 Conceptual overview of question text tagging.

3.2 Accuracy Matching and User Tag Generation

As we have mentioned above, the whole system can quickly get data to perform better and have a fair recommend effect at the very beginning. In order to deal with the user tags shortage, we introduced the practice accuracy matching technic. This accuracy can be initialized by user himself or overall questions average accuracy. The score function is showed in equation (1)

$$R_{matching} = 1 - ABS(a_{question} - a_{user}) \quad (1)$$

First recommendation based on this approach will be transported to the user in a rather smooth way. Then the user practice the recommended question and tell the system right-wrong about the result. Immediately, both historical accuracy will be updated based on this result so do the tags. For instance, if the result is right, the user tags will be fetched from question tags and marked as a positive weight. Otherwise, they will be marked as negative weight. More details of this process can be found in figure 3.2.1 which shows the overall pipeline. After cold-start and finished several rounds of recommendation, the number of user's tags will strikingly increase and corresponding weight will be allocated unevenly. The system has been warmed up and we can apply some more personalized recommendation.

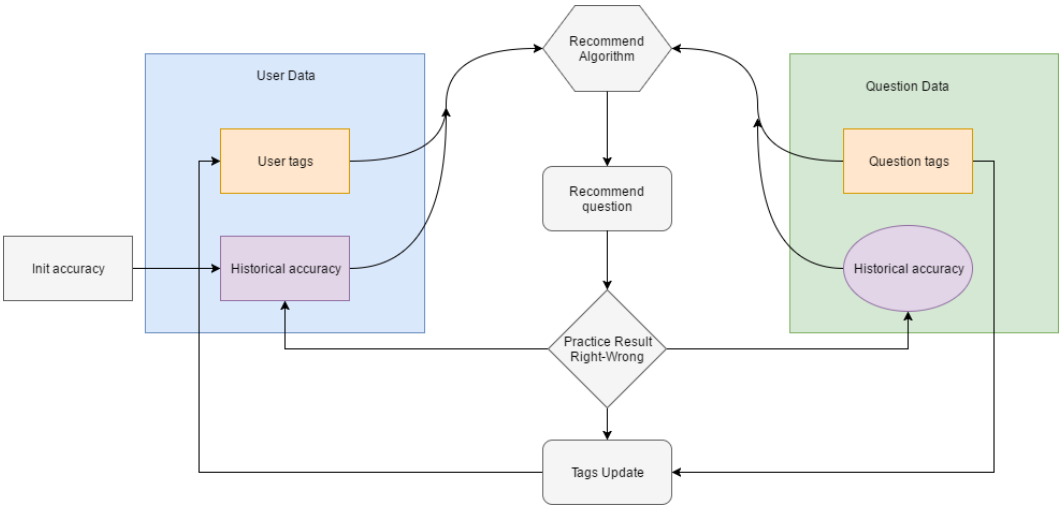

Figure 3.2.1 Conceptual overview of accuracy matching user tag generation pipeline.

3.3 Making Recommendations: Tag-Based Recommend

As showed in figure 3.2.1, we implement the recommend algorithm based on the common tags between user and questions.

ALGORITHM 1: Tag-based Recommend Algorithm
user_tags← *user*
question_set ← *overall questions*
question is inside *question_set*
for *question* ***is inside*** *question_set*, **do**
    common_tags ← *user & question tags in common*
    for ***each*** *tag* ***in*** *common_tags*, **do**
        question_score ← each question recommendation rate from *question_set*
        convert *weight* to *score*
            scale *score* by *weight / number_of_all_same_tag*
        question_score ← *question_score + tag_score*
    end
end
question_score ← *question_score + accuracy_score*
sorted *question_set* **by** *question_score*
return *question_set[0]*
end

Above is our tag-based recommend algorithm. As it shows, the overall algorithm focuses on the common tags between user and questions. Question score is first summed up by the scaled weight of common tags. In order to prevent the fact that some question may have too many tags and get unfair high recommending score, we scaled the weight by dividing the number of all the same name tags. Considered the initial state, we merged the score of matching rate from the last part of the

system. We finally achieve the evaluating score for each question in the question set. The score function is showed in (2).

$$S(\mathrm{q}) = R_{matching} * 100 + \sum_{tag}^{all\ tags} \left(\frac{W_{tag}}{N_{number\ of\ same\ tags}} \right) \quad (2)$$

$S(\mathrm{q})$ means the evaluated score for each question in question set, $R_{matching}$ is what we got from the accuracy matching process, W_{tag} represent the weight of each tag in common tags and $N_{all\ same\ tag}$ means the number of all same name tags. Noted that the more tags we have, the more personalized recommendation we would have. In other words, with the usage of the system, the recommendation of the system show increasingly personalization.

4 RESULTS AND DISCUSSION

4.1 Question Tagging Results of Verbal Reasoning Questions

Table 1. Some sample questions and their tags

Question-id	Accuracy	Auto-Tags
f2azxj	0.39	Clause, Adversative, Refer, Repeat
82b0xj	0.67	Refer, Repeat, Reverse
f2b1dj	0.22	Negative, Repeat, Refer
72b0yj	0.54	Positive, Repeat
b2b1nj	0.15	Positive, Negative, Repeat, Refer

Total 78 sample questions have been added into our system[2]. All of our test questions can be found on *http://gre.kmf.com/question/%s.html* (%s = question-id). Each question went through the auto-tagging process and has their tags showed in Auto-Tags column. Tags are generated by following pattern:

Table 2. Part of sample pattern used in tagging

Regular Expressions	Auto-Tags
r'.*who$'	'Clause'
r'.*:$'	'Repeat'
r'.*not$'	'Reverse'
r'.*this$'	'Refer'
r'.*dispute$'	'Negative'

From the auto tags result of each question, different questions' tags are in high degree of diversity and highly matching with the feature of the question context. These indicates that our process of NLP works good enough.

4.2 Recommendation Result of ETRS

We first let user who is preparing for GRE test practice in ETRS system 7 times. The result of his practice shows in Table 3. "T" means that the result is right, "F" means that the result is different from the official answer. For the 8th recommendation, the recommend list gained by our ETRS is

represented in Table 4. Part of questions in the question set have been showed in the sort tag-based recommendation algorism detailed in equation (2).

Table 3. results of one user in 7 practice

Question-id	22b0oj	22b23j	22b1xj	62b0pj	82b04j	f2azxj	02azzj
Right	T				T		T
Wrong		F	F	F		F	

Table 4. Recommendation List of one user after 7 times practice

Question-id	Auto-Tags	Recommend-score
c2b05j	Positive, Clause, Repeat	97.03
72b1zj	Refer, Repeat, Reverse	89.33
891jwk	Negative, Repeat, Refer	87.28
62b0qj	Positive, Repeat	77.31
72b1yj	Positive, Negative, Repeat, Refer	69.92

From the tables and results above, we can tell that our tag-based recommend algorism is not been trapped by the number of tags that a single question has and have a good diversity of tags and questions in recommendation list.

5 CONCLUSIONS

In this paper, we first analyze the structure of question text in text completion of verbal reasoning. Then we achieved auto tagging for questions text via nature language processing. After we get tags for question, we suggested an accuracy matching approach to add tags for users and warm up the whole system. Finally, with enough tags and tag-based algorithm, we made this system from end-to-end and performed an ETRS for verbal reasoning questions.

As for future work, text tokenization and question tagging can be more intelligent with new technic in NLP such as machine learning algorithm such as LSTM & Recurrent Neural Network. With more usage and user data, we can also try other recommendation algorithm like user-based or item-based filtering.

REFERENCES

[1] Educational Testing Service. The Official Guide to the GRE Revised General Test, 2nd Edition. McGraw-Hill Education.

[2] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J., 2001, April. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295). ACM.

[3] Bird, Steven, Edward Loper and Ewan Klein (2009) *Natural Language Processing with Python.* O'Reilly Media Inc.

[4] Segaran, T., 2007. *Programming collective intelligence: building smart web 2.0 applications.* O'Reilly Media, Inc.

[5] Cao, Y. and Li, Y., 2007. An intelligent fuzzy-based recommendation system for consumer electronic products. *Expert Systems with Applications*, *33*(1), pp.230-240.