ARTIFICIAL INTELLIGENCE IN MEASURING HIGHER ORDER THINKING SKILLS

Zalina Binti Jamalludin  
(zallyjam@gmail.com)  
Mohd Işam Bin Embong  
(isham2015@yahoo.com)

ABSTRACT

This study leverage student higher order thinking skills for multiple choice questions answering, and employ lightweight machine learning techniques to improve aggregation accuracy of student sourced answers to these questions. In order to develop more effective aggregation methods and develop them empirically, this study developed and deployed a system based on science subject. This study found that the students able to answer 90% of the questions correctly. But, the number decreased to 60% for the harder questions. However, to improve the success rate of these harder questions, this study investigated novel weight aggregation schemes for aggregating the answers that previously obtained from the students. By using weights optimized for reliability of participants, this study shows to pull up the accuracy rate for the harder questions by 15%, and to overall 95% average accuracy. The final results are to provide a show case for the benefits of applying machine learning techniques for building more accurate question answering system.

Keywords: multiple choice questions answering, aggregation accuracy, weights optimized

1. Introduction

Question answering (QA) is a computer science discipline within the fields of information retrieval (IR) and natural language processing (NLP) techniques, which is concerned with building systems that automatically answer questions posed by humans in a natural language. QA is studied as a fundamental problem by the artificial intelligence (AI) and the machine learning (ML) communities. Since search engines can answer well-performed factual queries successfully, their accuracy drops significantly for non-factual and natural language questions. The incorporating of IR and NLP has been a substantial research effort on answering this kind of questions. Since both of understanding and answering of the questions require solving complex AI problems, the achieving acceptable accuracy for QA still remains a hard problem (Aydin, Yilmaz, Li, Li, Gao, & Demirbas, 2014).

In this study, researcher leverage students’ higher order thinking skills (HOTS) to help AI for more accurate QA. The researcher considers students’ sourcing of QA and employ lightweight ML techniques such optimization with methods. HOTS refers to the ability to apply knowledge, skills and values in reasoning, reflection, problem solving, decision making, innovating and creating something new (Ministry of Education, 2013). The 21st century pedagogy inculcates HOTS elements to encourage deeper thinking activities among students (Sulaiman, Muniyan, Madhvan , Hasan, & Abdul Rahim, 2017) which in line with the aspiration of the Malaysian Education Blueprint 2013-2025. According to Nessel & Graham (2007), thinking skills is the most basic skills that can be developed in the classroom and is the key to high achievement for all students. The concept of HOT originated from the Bloom (1956) taxonomy of cognitive domain (Forehand, 2010), these cognitive domains involves knowledge and the development of
intellectual skills and hierarchically ordered from concrete knowledge to abstract (Pappas, Pierrakos, & Nagel, 2012).

Higher order thinking skill goes beyond memorizing and recalling facts and data. It even goes beyond comprehension. Higher order thinking skill refers to cognitive processes that involve analytical, critical or creative thinking. Bloom categorized intellectual behavior into six levels of thinking, knowledge, comprehension, application, analysis, synthesis and evaluation (Yahya, Toukal, & Osman, 2012). The first three levels of Bloom’s taxonomy require application, while the last three require students to use HOTS (Yahya et al. 2012; Forehand, 2010).

Science subject is selected because science experiences are an avenue where higher order thinking skills of analyses, synthesis and evaluation are applied by students in their learning process. Gradually, through that experience, students become problem solver, thoughtful decision maker and life-long learner because “higher order cognition helps them to become independent learners” (Sulaiman et al. 2017; Noor, 2008). This process enables students to incorporate the new knowledge with the existing ones for deeper understanding in a meaningful way. This is possible because ability to think impacts students’ cognition, achievement and attitude.

In order to simplify the problem, researcher focuses to multiple-choice question answering (MCQA), because for aggregation purposes it is more productive to ask MCQ than asking open-domain questions (OPQ). In contrast to OPQ which result in a large set of possible answers, MCQA limits the answers to predefined set of multiple choices, and this facilitates participation of the crowd, improves the answer ratio and latency, and simplifies the aggregation process. Researcher contends that providing MCQ is feasible for many applications, and it is also possible to automate the process of adding multiple choices to an OPQ (Lin, Mausam, & Weld, 2012; Aydin et al. 2014).

For this study, the development effective aggregation methods and how to evaluate them empirically are still an issue. However, researcher developed and deployed a simple crowd sourced system called “Test Me My Science Knowledge” (TMMSK) for students to answer. The idea is to develop an Android application to let the students answer the questions with their mobile devices. This idea is inspired by the IBM Watson’s success at Jeopardy and aims to utilize the students to answer TMMSK questions accurately. TMMSK app has been downloaded and installed more than 500 times and it has enabled the researcher to collect data about MCQA dynamics. Over the period of 3 months, the researcher collected less than 1 GB of MCQA data using Android app. In researcher’s dataset, there are 300 questions and 1000 unique answers.

Studying the data collected, researcher find that by just going with the most selected answer in the aggregation (majority voting - MV scheme), we can answer 92% of the questions correctly; however, these are mostly entry-level questions. In the application, there is a safe zone between 7th and 8th level questions, after which the questions become much harder. After question 7, the success rates of MV plunge quickly down to 60%. In other words, the correct answer is half of the time not the most popular answer for these harder questions. Ideally the final aggregated answer should lean towards the answers of capable rather than unreliable participants, but the ability of each participant is unknown a priori.

In this study, the researcher investigates how to improve the success rates of these harder questions. To this end, researcher investigates novel weighted aggregation schemes for aggregating the answers obtained from the application. Researcher proposes to integrate the process of estimating participant abilities and deriving correct answers in a unified model. The basic idea in the proposed scheme is that the correct
answer is obtained by a weighted voting among multiple participants where more reliable participants have higher weights. The participants who provide correct answers more often and more confidently will be regarded more reliable.

To enable this effective aggregation technique, researcher introduces two improvements. First, asks the participants how confident they are with their answers just after they choose an answer. Then, for optimizing participant’s results on harder questions, researcher uses lightweight ML techniques (Dai, Weld, and others 2011; Kittur et al. 2013). Researcher weights participants’ answers according to their performance and the confidence of their answers. Researcher proposes a novel method to estimate both participant weights and aggregated answers simultaneously. The method is inspired by HITS algorithm (Kleinberg 1999) for webpage authority and truth discovery (Yin, Han, and Yu 2007) for conflict resolving: The candidate answers from reliable participants should be weighted more in the aggregation. Then using the optimized co-variants researcher aggregates weighted answers in order to decide a final answer.

The results present the effectiveness of using participants’ confidence while answering the questions to solve this problem. The researcher able to pull up the accuracy rate for the harder questions by 15% over that of MV and to over 90% average accuracy, by using optimized weights for answers derived from participants’ confidence. These findings suggest that it is feasible to build the super player by aggregating answers from many players effectively using lightweight ML techniques. TMMSK application provides a good case for the benefits of applying AI techniques for building more accurate crowd sourced QA systems. In future work, researcher will investigate adapting lessons learned from the TMMSK application to general or location-based crowdsourcing applications and recommendation systems.

2. TMMSK Architecture

In this section, researcher proposes the TMMSK architecture application that enables the students to answer TMMSK with their smartphones. Inspired by the the idea of CrowdMillionaire Architecture (Aydin et al. (2014)), the overall architecture of TMMSK is shown in the Figure 1. TMMSK consists of three main parts, (1) an admin part for entering the questions and multiple choices, (2) a mobile side for presenting the questions to the students and letting them answer the questions, and (3) a server side for dispatching the questions, collecting the answers, providing statistics and running our MCQA algorithms
Figure 1: The system architecture

On admin side, the project members type the questions and multiple choices earlier in the system. The server aggregates the inputs to form the question and four choices and send them to the participants. The server collects input from admin machines, packages the question & the choices with some statistics and forwards them to all participants. Then, the participants select their answers along with their confidence levels for their answers, using their phones. Finally, the data is sent to the servers through the Client APIs and saved in the database.

Figure 2 shows a screenshot of the TMMSK mobile app user interface (UI). The participants are incentivized by the joy and the desire of the gameplay itself. To this end, they are forwarded to Statistics screen when the game is over. This component provides the list of the most successful 10 participants of the game, ranked based on the correct answer count and the response time. Note that, this ranking algorithm also takes the participants’ confidence choices into account: they gain or lose more points if they are right or wrong on their answers on which they were more confident.

Over the period of 3 months, researcher collected less than 1 GB of data using TMMSK. In the dataset, there are 300 questions and 1000 unique answers to those questions from TMMSK participants. The dataset includes detailed information on the gameplay: (1) how much time it took for a question to arrive to a participant, (2) when the question is actually presented to the participant on her device, and (3) when exactly the participant answered the question.
3. Methodology
This is a quantitative research that presents the methods to incorporate crowdsourcing and ML techniques in order to build TMMSK test. The objective is to collect the answers from the students, and then aggregate using ML techniques to accurately answer the questions. The proposed formulations are discussed below.

Problem Formulation: a question set $Q$, and each question in this set $q \in Q$ is answered by a set of students $Pq$. For a given question $q$, each student in the set $p \in Pq$ gives a candidate answer $x_{qp}$, and it can be one of the choices in set $S = \{A, B, C, D\}$. Our objective is to aggregate candidate answers among students, and get an aggregated answer $x^*_q$ for each question $q$.

3.1 Majority Voting (MV)
A naive solution for aggregating the candidate answers is to select the choice which is claimed by the majority of the participants. Recall that, each candidate answer $x_{qp}$ can be one of the choices in set $S$. Therefore in majority voting, for each question $q$, the highest voted choice by the participants is selected as the final answer $x^*_q$:

$$x^*_q = \arg \max \sum_{p \in Pq} 1(x_{qp} = x) \quad (1)$$

where $1(\cdot)$ is an indicator function, which outputs 1 when $x_{qp}$ is the same as $x$; otherwise, outputs 0.

3.2 Confident Only Voting (CO)
Each candidate answer $x_{qp}$ has a corresponding confidence label $c_{qp}$ attached. Namely, when a student provides an answer $x_{qp}$, she is asked to indicate how confident her answer is, and this confidence label $c_{qp}$ is attached to her answer. A student cans define her confidence level using one of our predefined confidence labels which are \{“certain”, “guessing”, “no idea”\}.

Researcher proposed to filter the collected data by only choosing $x_{qp}$, which has a confidence label “certain”. To this end, for each question $q$ we define a student set $P$
certain $q$ in which each candidate answer $x^p_q$ has a confidence label $c^p_q$ equals to “certain”. Then, we run the majority voting on this subset to decide the aggregated final answer $x^*_q\mathbf{x}$ for each question $q$:

$$x^*_q = \arg \max_{x \in S} \sum_{p \in P_{certain}} 1(x^p_q = x)$$  \hspace{1cm} (2)

### 3.3 Confident Weighted Voting (CW)

The aforementioned two algorithms are unweighted in the sense that students have equal weights when aggregating their answers. In the following, researcher proposes several weighted aggregation strategies. As the basic weight covariant, researcher uses the answer confidence value $c^p_q$ defined above. Instead of eliminating the candidate answers which are not labelled with “certain” as in the above method, here researcher set weights $w^p_q$ according to the confidence labels $c^p_q$. Her insight here is to give higher weights to the answers of students who are more confident and lower weights otherwise as follows:

$$x^*_q = \arg \max_{x \in S} \sum_{p \in P} w^p_q 1(x^p_q = x)$$  \hspace{1cm} (3)

Where $w^p_q$ is set according to the confidence label $c^p_q$ by this rule: for confidence labels “certain”, “guessing” and “no idea”, researcher sets the weights to be 3, 2 and 1 respectively.

### 3.4 Student-Mine Voting (SM)

In TMMsK test scenario, it is natural to assume that some students perform better than others. In order to leverage this fact, researcher introduces the concept of students’ weights, denoted as $w_p$. Note that; $w_p$ is different than $w^p_q$ as it is not specific to a question. In this algorithm, if a participant’s weight $w_p$ is high, the candidate answers from her will be more trustful. Researcher can incorporate such weights into the aggregation using weighted combination. Thus, in a weighted aggregation scheme, researcher trust more in the candidate answers $x^p_q$ that are provided by the students with better performance.

In this scheme, researcher should figure out how to link student weights with their performance. To tackle this challenge, researcher proposes a novel optimization method to estimate both student weights and aggregated answers simultaneously. The basic principle is inspired by HITS algorithm (Kleinberg 1999) for webpage authority and truth finding (Yin, Han, and Yu 2007) for conflict resolving: The candidate answers from reliable students should be weighted more in the aggregation, and the students who provide correct answers should be assigned higher weights. Based on this principle, researcher formulates the task as the following optimization problem:

$$\min \{x, \{wp\} \mid \sum_{q \in P} \sum_{p \in P_q} w_p \cdot d(x^p_q, x) \text{ s.t. } w_p \geq 0, \sum_p w_p = 1, \}$$  \hspace{1cm} (4)

where $d(\cdot)$ is a distance function which measures the difference between $x^p_q$ and $x$. In order to regularize the students’ weights and prevent them to be infinity researcher set $\sum_{p \in P_q} w_p = 1$ as a constraint function.
From Eq. (4), we can see that there are two sets of unknown variables, i.e., \( \{x\} \) and \( \{wp\} \). Therefore, a natural way to solve this optimization problem is to use block coordinate descent techniques (Bertsekas 1999). Specifically, researcher adopts a two-step procedure which iteratively updates one set of variables to minimize the objective function while keeping the other set of variables unchanged.

As researcher proposed solution is an iterative procedure, it may lose some information if researcher assumes that at each step only one candidate answer is true. To avoid this disadvantage, researcher represents candidate answers and aggregated answers by index vectors, which allow her to consider all possible values during the iterative procedure. For example, candidate answer “A” can be coded as \((1,0,0,0)\), “B” is \((0,1,0,0)\), and so on. In this case, researcher can use L2-norm function for the distance function \(d(\cdot)\) to measure the distance between two index vectors.

3.5 Confidence-Mine Voting (CM)

Researcher proposes another method to combine Confidence Weighted Voting (CW) and Student Mine Voting (SM) to utilize the power of weighted aggregation. The idea is simple: on top of the SM method, researcher incorporates the confidence labels into the index vectors. For example, consider a student who provides a candidate answer “A”. Then, based on the confidence label attached to her answer, the index vector is defined as follows:

If her answer has a confidence label “certain”, then the index vector will be \((1,0,0,0)\). If the confidence label is “guessing”, then the index vector will be \((\frac{1}{2}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})\). Finally, if the confidence label is “no idea”, then the index vector will be \((\frac{1}{3}, \frac{2}{9}, \frac{2}{9}, \frac{2}{9})\). For the candidate answers “B”, “C” and “D”, they will be transformed into the index vectors in similar way. Then running our optimization algorithm defined in Eq. (4) on these vector values, researcher calculates a weight \(\{wp\}\) for every student \(p \in P\). Finally based on these calculated weights, researcher chooses an aggregated answer \(x_q^*\) for every question \(q \in Q\) similar to the previous SM method. Thus, the only difference between SM and CM is the input vectors that researcher uses to initiate the optimization.

4.6 Bing Fight

In order to compare the performance of these methods with the search engine based QA systems, researcher implemented the algorithm defined by (Lam et al. 2002). Despite the authors stated Google performs better, researcher build the system based on Bing, because Google started to limit the number of queries for automated search requests.

In the naive counting method, researcher produces the search engine query string by concatenating each choice with the question. Therefore, for each question in the dataset, researcher has four different search engine queries. Researcher searches these four queries on Bing, and calculates the total number of result pages, i.e. number of hits, for each of them. The final answer of this method is the choice with the maximum number of hits. Note that, researcher do not make any query modifications suggested in Distance Score (Lam et al. 2002) algorithm such as ignoring some buzz words or decoupling phrases of the question and choices, because they are not in the scope of our work. We define \(B(\cdot)\) as the function that returns number of result pages i.e. number of hits for a query string. In the following equation, string \(q\) denotes the question text as string \(x\) denotes a choice string, which is different for each of the four \(x \in S\):
\[ x \ast q = \arg \max_{x \in S} B(stringx||stringq) \]  

4. Results and Evaluation

In this section, researcher evaluates her methods described using her TMMSK data. Table 1 shows how each of these methods perform by question level. As shown in Figure 3, researcher does not have statistically significant number of questions from level 11 and 12 in her dataset. Thus, researcher does not include the results for those questions as they are not enough to accurately measure the performance of her algorithms. Note that, for any of the algorithms, if more than one choice has the maximum vote, then researcher counts it as undecided. When calculating the accuracy of the algorithms, researcher counts the undecided questions as failure (i.e. not as success).

Researcher first run the Bing Fight on her question set. Researcher directly queried via TMMSK questions. It can answer 30.50% of all questions successfully, that is a little less than the expected performance according to the previous works (Awadallah and Rauber 2006; Lam et al. 2002). This performance loss is due to several reasons such as: (1) the undecided questions, (2) using Bing instead of Google, which might perform better and (3) not making the small query optimizations which are mentioned in the related work too. Ignoring the undecided questions, overall success rate increases to 37.06% which is closer to the results in previous works (Awadallah and Rauber 2006; Lam et al. 2002).
Using the MV algorithm, i.e. the elemental crowdsourcing algorithm, the participant is able to answer 95% of the earlier or easier questions correctly. Although the MV is successful enough on those easier questions, its success rate plummets on the higher level questions as seen in the second column of Table 1 and in Figure 4. These results also represent an interesting fact about the format of the game: the success rates of the MV decrease more than 20% between the 7th and the 8th questions, because the hardness of the questions increase a lot after the safe zone of the game. This distinctive change in the performance of participants enables the researcher to clearly categorize easier and harder questions based on question level. Figure 4 shows how the MV algorithm and Bing Fight performs by question level. It is clear from the graph that even the elemental participants algorithm significantly outperforms the Bing Fight.

![Figure 4: Bing Fight vs Majority Voting](image)

Regardless of whether they use the confidence data or not, all of the methods are able to answer roughly 95% of the lower level questions correctly. Namely, failure is a rare case. On those lower level questions, the CW or CO methods are able to answer some of the questions which the MV fails on, but on the other hand the CW or CO methods fail on some other questions which the MV answers correctly. However, when it comes to the higher level questions (i.e. 8th question and above), even the basic confidence-based methods seem to be more successful than the MV as seen on the Figure 5. Henceforth, the discussions will focus on the harder questions.

![Figure 5: Majority Voting vs Confident Only Voting and Confidence Weighted Voting](image)
As it is clear from the Figure 5, the CO method slightly outperforms the CW method on the higher level questions. On the other hand, CW is a more robust algorithm as it is less dependent on the size of data. Although the data is big enough and researcher did not observe any problems caused by small data size while running our tests, picking only the “certain” labelled answers can reduce the data size significantly. In some other domains, this reduced data size may cause problems, such as fluctuation and speculation on final answers as they may be dominated by a small number of faulty participants. Therefore, using the CW and the CO together in a hybrid fashion would be more dependable when the data size is not big enough. In that hybrid approach, CO function can lead the final decisions and CW can be used when CO cannot decide comfortably. That way, first only the “certain” answers will be counted, and in case of a close result then the hybrid algorithm will decide considering other answers too.

Figure 6 reveals that the optimized weight based methods perform significantly better than all the other methods, especially on the harder questions. On those harder questions, the PM method can successfully find the correct answers, for half of the situations where majority of the crowd fail. Then by incorporating the confidence data to the optimization, the CM method performs above 90% for all the question levels. This great improvement indicates using a lightweight training on the crowdsourced data, researcher are able to build a super-player for the TMMSK game.

![Figure 6: Majority Voting vs Participant Weighted Voting and Confidence Mine Voting](image)

5. Conclusion
In this study, researcher presents crowdsourced methods using lightweight ML techniques to build an accurate MCQA system. The accuracy of her MCQA methods are literature promising. By using optimized weights for answers derived from students’ confidence, researcher builds a TMMSK test that can answer the questions from all difficulty levels with an accuracy of above 90%. In future work, researcher will implement the systems and investigate adapting lessons learned from recommendation systems.
References


