An Automatic Short Answer Correction System Based on the Course Material

Zeinab Ezz Elarab Attia1* Waleed Arafa1 Mervat Gheith1

1 Institute of Statistical Studies and Research, Cairo University, Egypt
* Corresponding author’s Email: eng.zeinabezz@gmail.com

Abstract: The paper presents a new system for correcting short answer questions using the course material in addition to the model answer. Such system overcomes the limitations of the current systems that are just based on calculating the similarity degree between the model answer and students’ answer. Thus they neglect any correct answer that may be written by a student but not mentioned in the model answer. Also, they neglect correcting student answers that contains definition or explanation for a concept stated in the model answer. To overcome such limitations, the proposed system uses the course material to expand the model answer with more correct answers via answering the short answer questions. In addition to expanding the model answer’s concepts with their synonyms and definitions (using is-a relation and have relation). The proposed system is tested and evaluated on Texas dataset. The proposed system outperforms those systems that are evaluated on the same dataset, by achieving a correlation value about 0.8.

Keywords: Automatic short answer correction system, Automatic correction systems.

1. Introduction

Examination is a successful practical method for evaluating students’ educational level. Recent studies proved that, continuous exams improve students’ educational level when they are corrected and feedbacks are sent to students immediately. Unfortunately, this task is tedious and time consuming. Thus this issue figures out the importance of having automatic correction systems that help instructors correcting exams automatically. Several automatic correction systems are proposed in the literature that are categorized according to the question's type, e.g., fill in the blank, essay, short answer questions...etc.

The paper is concerned with correcting the short answer questions automatically. Unfortunately, the current automatic short answer correction systems suffer from:

1. Comparing student answers only to the model answer. As the result, they:
   i. Neglect any correct answer that is not mentioned in the model answer.

Example:

Question: State the advantage of Object Oriented Programming?

Model answer: reusability and modularity

Student answer: inheritance and polymorphism

ii. Neglect correcting students’ definitions for concepts that are stated in the model answer.

Example:

Question: Where are variables declared in C++?

Model answer: local variable declared inside a function and global variable declared outside the functions.

Student answer: they can be declared globally outside functions also they can be declared locally inside a function.

Global variable definition

Local variable definition

2. In addition to lacking a standard dataset to test their system on it. Thus each system is tested on its own dataset.

Thus the paper presents a proposed ontology based correction system that deals with such limitations via using the model answer, the
Table 1. Comparison between automatic short answer correction systems

<table>
<thead>
<tr>
<th>Correction System</th>
<th>Algorithm</th>
<th>Applied domain</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Systems based on Similarity Measure algorithms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Texas [1]</td>
<td>Calculate the similarity degree between both answers using knowledge-based and corpus-based similarity measures.</td>
<td>Computer science (Texas dataset)</td>
<td>Best correlation 'Jiang &amp; Conrath '0.44' - LSA '0.4628'</td>
</tr>
<tr>
<td>Gomaa &amp; Fahmy [2]</td>
<td>Calculate the similarity degree between both answers using string-based and corpus-based similarity measures.</td>
<td>Computer science (Texas dataset)</td>
<td>Best correlation '0.5' n-gram then Disco1</td>
</tr>
<tr>
<td>Pribadi [3]</td>
<td>Calculate the similarity degree between both answers using the Cosine Coefficient. Thus, it cannot score answers with different words and similar meaning</td>
<td>Computer science (Texas dataset)</td>
<td>Less than 0.4</td>
</tr>
<tr>
<td>Pado [10]</td>
<td>Four stages: 1. Lemmatize both answers. 2. Remove stop words. 3. Remove all lemma words from both answers that appear in the question text. 4. Calculate the similarity degree between both answers using DKPro Similarity implementation of Greedy String Tiling.</td>
<td>Computer science (German language)</td>
<td>78%</td>
</tr>
<tr>
<td>Systems based on Structure Matching</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IndusMarker [5, 8]</td>
<td>Two stages: 1. Represent model answer in QAML, 2. Analyze student answer text</td>
<td>Object oriented programming</td>
<td>69.4%</td>
</tr>
<tr>
<td>Systems based on NLP techniques</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Systems based on Machine Learning Techniques</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Noorbehbahani [9]</td>
<td>Introduces M-BLEU that: 1. uses a domain dictionary, 2. associates each n-gram with a weight respecting its importance, 3. Calculate similarity score with the shorter correct answer with the maximum M-BLEU.</td>
<td>Computer and information engineering</td>
<td>Obtained correlation ‘0.85’</td>
</tr>
</tbody>
</table>

question, and the course material to correct a student answer. The benefit of using course material is to:

1. Expand the model answer with more correct answers that can be extracted using the question with the course material. (resolve the limitation ‘1.i’)
2. Expand each concept in the model answer with its definition. (resolve the limitation ‘1.ii’)

However, to overcome the second drawback, the proposed system is evaluated on an already exist dataset namely, texas dataset. Texas dataset is used to evaluate another three systems [1], [2], [3]. Thus, the proposed system’s results are compared with them.

The paper is organized as follows; section 2 presents some of the current short answer correction systems. The proposed model is presented in details in section 3. Evaluating the proposed model is discussed in section 4. Section 5 concludes the paper.

2. Short answer correction systems

This section provides some of the short answer questions correction systems namely, Texas [1], Gomaa [2, 4], Pribadi [3], IndusMarker [5], C-rater [6], Automarking [7], Noorbehbahani [9] and Pado [10]. Table 1 classifies such systems according to the used technique and compares them.

3. The proposed ontology-based short answer correction system

3.1 The system input

The system takes as its input: a course material, a model answer, a student answer, and a question text. The course material is used to build a domain ontology. Such ontology is used to add more correct answers through expanding the model answer, and answering some types of questions (only definition and comparison questions).
The ontology is built by using an ontology learning tool with the aid of the course instructor for reviewing the built ontology.

The model answer is expanded using the course material to correct the student answer. Some types of questions (namely, definition and comparison questions) are also used to expand the model answer via answering them automatically using the course material. All such expansions are combined together to create the complete model answer.

3.2 The system used tools

- To build a domain ontology, the text2onto tool is used on the course material to extract the domain main concepts associated with the relation between them.
- To replace a pronoun to its referred noun, a coreference tool is used, Stanford CoreNLP.
- To lemmatize answers, the Stanford CoreNLP is used.
- To determine the part of speech, a POS tagger is used.
- To stem answers, the Paice/Husk algorithm is used since, Paice [11] proved that the over-stemming algorithms increase the recall than the under-stemmers. This guarantees matching the related words. As according to Moral [12], Paice/Husk has higher performance than the other over-stemmers.

3.3 How does the system work?

The student answer is scored via calculating its similarity degree with the extracted complete model answer predicates. The scoring method is divided into four phases namely: the preprocessing phase, key-answer extraction phase, complete model answer extraction stage, and grading phase.

1st: Preprocessing phase

The preprocessing phase is applied on both the model and the student answers. In this phase, the model answer is prepared to extract the key answer from it. Whereas the student answer is prepared to match it with the complete model answer predicates (will be extracted in the third stage), see example 1, 2, 3, 4.

For each answer, apply the following stages:

1. Resolve pronouns in the answer, see example 1.

2. Divide it into a set of sentences using “,” “;”, “and”, “or”, “but”, “while”, “whereas”, “thus”, “so”, “however”, “normally”, “therefore”, etc. For example, see examples 2, 3.

3. For each sentence, chunk it into a set of phrases using a chunking tool, see examples 2, 3.

4. For each phrase, remove determiners (if exist), lemmatize the phrase and then stem it, see examples 2, 3.

5. As depicted in example 3, applied on the student answer only,
   - If it contains concepts that match the following tregexPP $-\text{NP}_1 \text{ $+ \text{NP}_2}$
     Such that either:
     - Both \text{NP}_1 and \text{NP}_2 are found in the ontology and there is a relation between them
     - OR, [\text{NP}_2 \text{ NP}_1] is a concept in the ontology.
     Then, replace them with: [\text{NP}_2 \text{ NP}_1 ]

6. For each sentence contains a connector (i.e., but, however, while, etc), resolve the following connector problem, if exist:
   - if both phrases have a common verb and one of them ended with that verb, complete such phrase using the other completed phrase, see example 3.

Example 1: (Resolving pronouns on a student answer)

Constructors cannot return values, so they cannot specify a return type. Normally, constructors are declared public.

After resolving pronoun:

Constructor cannot return values, so Constructor cannot specify a return type. Normally, constructors are declared public.

The End of Example 1

Example 2: (Applying the preprocessing phase on a model answer for a question)

A function prototype includes the function signature. The function definition includes the actual body of the function.

Pronoun resolution (Step 1): Not applied here.

After sentence division (Step 2): S1 \rightarrow A function prototype includes the function signature.

S2 \rightarrow The function definition includes the actual body of the function.

Sentence#1: A function prototype includes the function signature.

Phrases (Step 3): [NP: A function prototype][VP: includes] [NP: the function signature]

Stem (Step 4): Lemma: [NP: function prototype] [VP: include][NP: function signature].
Function prototype only includes the access function name.

Phrases (Step 3): [NP: The function definition] [VP: includes][NP: the actual body][PP: of][NP: the function]

Phrases (Step 4): [NP: Function definition] [VP: includes][NP: the actual body][PP: of][NP: function].

Step 5: Replace the concept “cod for func” with the concept “func cod” as it matched the tregex PP -$NP $+NP. Also, both NPs are found in the ontology and there is a relation between them.

Then, Func defini include func cod to perform func activ.

The End of Example 3

Example 4: (Resolving the connector problem on a student answer)

The data member can be accessed outside the class while the local variable cannot.

Step 7: The data member can be accessed outside the class while the local variable cannot be accessed.

The End of Example 4

2nd: Key-Answer Extraction phase

The key-answer extraction phase is applied on the model answer using the ontology via extracting the range, relation and domain from each sentence in it, see example 5. The phase is applied as follows:

For each sentence in the model answer:
1. Extract the concept that represents a range from the sentence using the ontology
   (a) If the sentence phrases contains a concept that matches the tregex: PP -$NP$+NP$+NP$Such that, both NP1 and NP2 are found in the ontology. Also, there exist a relation between NP1 and NP2.

   Then, NP1 (that is related to NP2) is the range.

   For example: “local variable inside function”. The range will be “local variable”.

   (b) If the sentence phrases contains a concept that matches the tregex: PP -$NP$+NP$+NP$Such that, NP2 NP1 is found in the ontology.

   Then, NP2 NP1 is the range.

   For example: “type of parameter”, The range will be “parameter type”. “Location in memory”, the range will be “memory location”.

The End of Example 4
(c) If the sentence phrases contains a concept that matches the Tregex NP such that NP is found in the ontology. For example, local variable, fuzzy ontology, inheritance, polymorphism. 
Then, such concept is the range.

(d) If the sentence phrases contains a concept that matches the Tregex NP such that a subset from NP is found in the ontology. 
Then, the only matched subset is the range
For example: “word parameter”. The range will be “parameter”.

(e) If the sentence phrases contains a concept that matches the Tregex NP such that more than one subset from the NP is found in the ontology. 
Then, the 2nd subset (that is related to the 1st subset) is the range.
For example: class data member (both “class”, “data member” are found in the ontology). The range will be “data member”.

2. Extract the range relation from the sentence using the ontology such that:
   • If the sentence is positive,
     For the extracted range, match its ontology relation or one of its synonyms. If found, then, the relation is found
   • If the sentence is negative,
     For the extracted range, match its ontology opposite relation. If found, then, the relation is found

3. Extract the domain from the sentence (if exist) using the ontology
   (a) If the sentence contains a concept that matches the tregex PP $-NP_1$ $+NP_2$ such that, NP1 or NP2 or subset from any one of them is related to the extracted range with the extracted relation.
   Then, the related one is the domain.
   For example: “data member is accessed by member function inside class”, see Fig. 1.
   The Range is “data member”
   Relation is “accessed”, domain is “class”.

   (b) If the sentence contains a concept that matches the tregex PP $-NP_1$ $+NP_2$ such that, NP1;NP2 is found in the ontology to be related to the extracted range with the extracted relation.
   Then, NP1;NP2 is the range.
   For example: “variable is a location in memory”, see Fig. 2.
   The range: “variable”, relation: “is a”
   The domain is “memory location”.

(c) If the sentence contains a concept that matches the tregex NP such that NP or a subset from it is related to the extracted range with the extracted relation.
Then, the related one is the domain.

4. Construct the sentence predicate from the extracted range, relation and domain (if exist):
   • if the range, relation and the domain are all exist, the predicate will be:
     relation (range, domain); for positive sentence
     not relation(range, domain) for negative sentence
   • if only the range is exist(range)

Datum member
Access (inside)?
Class
Figure. 1 Example on extracting range using ontology
Variable
Is a
Memory location
Figure. 2 Example on extracting range using ontology

Example 5: (Applying the key-answer extraction phase on the model answer of the same question)
Sentence#1: [NP: func prototyp][VP: includ] [NP: func signat]
Domain: “func signat”
The predicate:
includ (func prototyp, func signat)
Sentence#2: [NP: func defini][VP: includ] [NP: actu bod][PP: of ] [NP: func].
Domain: “func bod”
The predicate:
  includ (func defini, func bod)

The End of Example 5

3rd: Complete Model Answer Predicate Construction phase

The complete model answer construction phase constructs the complete model answer from both the question and the model answer’s predicates, see example 6. This phase is applied as follows:
1. Answer the question text then from the answer construct the predicate(s) of the question answer(s)
   i. extract the concept to be answered via:
      a. For comparison questions, remove concepts matching the regular expression “[difference| similarity] .* between” or their synonyms.
      b. Chunk the question text into of phrases.
      c. For each noun phrase (NP) and (PP),
         • Remove the question headword, e.g., “What”, “Where”, “How”.
         • Remove determiners (if exist).
         • Stem them after lemmatizing.
         • Extract the concept to be answered, using the same criteria of extracting the model answer range(see step(1) in the 2nd phase)
   ii. For each extracted concept, answer the question as follows:
      • For definition questions, use the ontology to expand the concept with all its direct relations.
      • For comparison questions,
        - Similarity questions:
          Expand all concepts with domains and relations that they both have.
        - Difference questions:
          Expand all concepts with domains (or/ and) relations that are found in one of them and not found in the others.
      • All the predicates of the question answers will have the form: relation (concept, domain) where, the concept is the extracted concept, the relation and the domain are that extracted from the ontology
   iii. For each predicate in the question answer,
      • expand its range, relation, domain with its synonyms
      • expand its range, relation, domain with its ‘have’ relationship
      • if the predicate’s relation is positive
        - expand it by adding ‘not’ and the relation opposite.
      • if the predicate’s relation is negative
        - expand it by adding only the relation opposite.
2. Expand the instructor’s model answer (phase 1):
   • For all types of questions other than the comparison and the definition questions,
     - Expand each predicate in the model answer with domains having the same relation and range in it.
3. For all questions other than the comparison questions,
   • Remove the question concept and/or relation from each predicate in the complete model answer, if exist.
4. Expand the instructor’s model answer (phase 2):
   • For each predicate in the model answer,
     - expand its range, relation, domain with its synonyms
     - expand its range, relation, domain with its ‘have’ relationship
     - if the predicate’s relation is positive:
       *Expand it by adding ‘not’ and the relation opposite
     - if the predicate’s relation is negative:
       *Expand it by adding only its opposite
5. Construct the complete model answer predicates:
   i. ORing each predicate in the model answer with its expansion ones
   ii. ANDing each predicate in the model answer with each other and with each predicate in the question answer
   iii. ORing predicate in the question answer with its expansion
6. For all the complete model answer predicates,
   • If both the domain and the range contain more than one word, such that there is a common word between them, then remove it from the domain of all complete model answer’s predicates.
7. For all comparison question,
If the concepts to be compare (range in the predicate) contain more than one word, such that there is a common word between them, remove it from each predicate in the complete model answer.

Example 6: (constructing the complete model answer predicates from the same question and its model answer)

1. Answer the question text
   What is the difference between a function prototype and a function definition?

   i. **Concept extraction**
      a. **Removing the RE**: What is the a function prototype and a function definition?
      b. **Phrases**: [NP What] [VP: is] [NP: function prototype] [NP: function definition]
      c. **Lemma NP, PP**: [NP: function prototype] [NP: function definition]
      **Stem NP, PP**: [NP: func prototyp], [NP: func defini]

   d. **Extract concepts to be answered**
      **Concept1**: “Func prototyp” WHY?? As the concept in the NP exist in the ontology
      **Concept2**: “func defini” WHY?? As the concept in the NP and exist in the ontology

   ii. **Question Answering**, see Fig. 3
      **Concept1**: declar (func prototyp, func)
      **Includ(func prototyp, function signat) not includ (func prototyp, func bod)

      Func prototyp
      Declar
      Func
      Func signat

      Func bod=func cod
      Not includ (func prototyp, func cod)
      declar (func defini, func cod)
      paramet=argu
      hav (func prototyp, function argu)

      **Relation synonyms:**
      declar= defin
      defin (function prototyp, func)

   2. **Expanding the model answer predicate**
      (phase 1) Not applied on comparison questions

   3. **Remove the question’s concepts and relation common word if exist**: Not applied on comparison questions

   4. **Expanding the model answer predicate**
      (phase 2) Since, the model answer predicate is already a subset from the question answer predicate, then its expansion will be included in the question answer predicates’ expansion.

   5. **Complete model answer construction**
      [[includ (func defini, func bod) OR includ (func defini, func cod)]
      AND
      [declar (func prototyp, func) OR defin (func prototyp, func)]
      AND
      [not includ (func prototyp, func bod) OR not includ (func prototyp, func bod)]
      AND
      [includ(func prototyp, func signat)OR hav (func prototyp, func nam) AND hav (func prototyp, func bod) AND
      hav (func prototyp, func return typ)]]

   6. **Remove the common word from the domain**
      [[includ (func defini, bod) OR includ (func defini, cod)]
      AND

---

*Figure. 3 Example on answering the question using ontology*
Let the total number of sentences in the instructor’s model answer is $x$. The total grade of the given question as assigned by the instructor in $G$. $G/x$ is the grade of each predicate in the complete model answer. Final grade is the student final grade.

$$\text{Final grade} = \sum_{i=1}^{n} SD_i \quad \text{eqn.}(1)$$

SD$_i$ value is calculated using the following algorithm as follows:

**CalSim(SA$_i$, MA$_i$)**

1. If (MA$_i$ format is Range) AND SA$_j$ matches it
2. SD$_i$=G/x
3. Else If (MA$_i$ format is Rel(Range)) then
4. \hspace{1cm} If (SA$_j$ matches it)
5. \hspace{1cm} SD$_i$=G/x
6. \hspace{1cm} Else If (Rel or Range is missed from SA$_j$)
7. \hspace{1.5cm} SD$_i$= 0.7*G/x
8. \hspace{1cm} Else
9. \hspace{1.5cm} SD$_i$= 0
10. End if
11. Else If (MA$_i$ format is Rel(Domain, Range))
12. \hspace{1cm} If (SA$_j$ matches it) then
13. \hspace{1.5cm} SD$_i$ =G/x
14. \hspace{1cm} Else If (Domain, Rel or Range is missed)
15. \hspace{2cm} From SA$_j$ then
16. \hspace{2.5cm} SD$_i$= 0.7*G/x
17. \hspace{1cm} Else
18. \hspace{2cm} SD$_i$= 0
19. End if

**Example 7: (Applying the grading phase on the preprocessed student answer for the same question)**

The question grade is: 5
Total number of sentences in the model answer: 2
Each sentence grade is: 5/2=2.5

**Complete model answer predicate**

**predicate# 1:**

includ (defini, cod) and its expansion  
matches student sentence # 3  
SA$_j$: Func **defini include cod** for func to perform Func activ.

**Then,** Sim_deg=2.5

**predicate# 2 & predicate #3:**

declar (prototyp, func) and its expansion  
not includ (prototyp, bod) and its expansion
Not matched with any student answer sentence

Then, Sim_deg=0

predicate# 4:

[includ (prototyp, nam) AND includ (prototyp, return typ) AND includ (prototyp, paramet)]

and its expansion

Part from it matches student answer sentence #1

AND sentence #2

SA1: Func proto typ onl includ access func name.

SA2: Func proto typ onl include paramet.

Then, Sim_deg=1.6

Final grade=5

The End of Example 7

4. Evaluation

4.1 Dataset

The proposed system is evaluated and tested on Texas dataset1. It consists of ten assignments with between four and seven questions each. Also, it includes two exams with ten questions each. These assignments/exams were assigned to an introductory computer science class at the University of North Texas. Each assignment includes the question, instructor answer, and set of student answers with the average grades of two annotators included. Moreover, the dataset includes the course material (.ppt) files. Note that this dataset is used to evaluate and test another two short answer question correction systems, which are Texas [1] and Gomaa et al [2].

4.2 Dataset usage

The dataset is used to evaluate the proposed system’s third module, the text similarity module. The course material is used to construct the domain ontology via using the text2onto tool. Then, the constructed ontology is reviewed by the course instructor. All the regular expressions used in the ontology are added by the course instructor. The domain dictionary is constructed manually by a domain expert.

4.3 Results and discussion

To evaluate the proposed system, Pearson’s correlation coefficient is measured against average human grades. Afterwards, it is compared with Texas [1], Gomaa [2], and Pribadi [3] systems as they are tested on the same dataset. Texas reaches the correlation values 0.328, 0.395, and 0.281 by applying LSA, ESA, and tf*idf respectively. However, Gomaa [2] reaches the correlation value 0.504 by combining String-based and Corpus-based similarity in an unsupervised way. On the other hand, the correlation value of Pribadi [3] is less than 0.4. The proposed system raises the correlation results to be 0.8001, see Fig. 5.

Pearson’s correlation measures the agreement degree between the human and the computer grades. Thus the higher the correlation value is, the more accurate the system is. From Fig. 5, the proposed system outperforms others as the result of using the course material for:

- expanding the model answer with more correct answers generated from answering the question.
- expanding the model answer with the definition of its contained concepts (using ‘is-a’ and ‘having’ relations).

Thus, this approach mimics the human thinking in detecting and correcting students’ correct answers that are not mentioned in the model answer. On the other hand, the other systems have low correlation value as they just calculate the similarity degree between a student answer against the instructor model answer without considering more correct answers.

5. Conclusion

In order to have an intelligent short answer correction system that mimics the human correction, the course material of the underlying exam is used. Via the course material, the system expands the model answer with the definitions of its contained concepts (using the ‘is-a’ and ‘have’ relations) and with more correct answers extracted by answering the question to correct a student answer.

https://web.eecs.umich.edu/~mihalcea/downloads.html

1 "Texas" dataset is available at [https://web.eecs.umich.edu/~mihalcea/downloads.html](https://web.eecs.umich.edu/~mihalcea/downloads.html)
Our future work is to represent the meaning of both answers using the Abstract Meaning Representation. Then calculate the similarity degree between them.

References