Automatic Answer Checker using Artificial Intelligence

Mr. Ravi Kumar B.N [1], Rishit Bansal [2], Vinayak. O. Hamsagar [3], Karan Ballal[4]
[1] Assistant Professor Dept of CSE
[2, 3, 4]Dept of CSE
[1, 2, 3] BMIST&M

Abstract- *An automatic answer checker application that checks and marks written answers similar to a human being. This software application is built to check subjective answers in an online examination and allocate marks to the user after verifying the answer. The system requires you to store the original answer for the system. This facility is provided to the admin. The admin may insert questions and respective subjective answers in the system. These answers are stored as notepad files. When a user takes the test, he is provided with questions and area to type his answers. Once the user enters his/her answers the system then compares this answer to original answer written in database and allocates marks accordingly. Both the answers need not be exactly same, word to word. The system consists of in-built artificial intelligence sensors that verify answers.*

Keywords- Answer Checker; Word Similarity; Sentence Similarity; Word order similarity; Synset; Text extraction; keyword similarity.

I. INTRODUCTION

An automatic answer checker application that checks and marks written answers similar to a human being. This software application is built to check subjective answers in an online examination and allocate marks to the user after verifying the answer. The system requires you to store the original answer for the system. This facility is provided to the admin. The admin may insert questions and respective subjective answers in the system. These answers are stored as notepad files. When a user takes the test, he is provided with questions and area to type his answers. Once the user enters his/her answers the system then compares this answer to original answer written in database and allocates marks accordingly. Both the answers need not be exactly same, word to word. The system consists of in-built artificial intelligence sensors that verify answers and allocate marks accordingly as good as a human being.

Automatic assessment of subjective answers requires Natural Language Processing based evaluation and automated assessment. Various techniques used are Ontology, Semantic similarity matching and Statistical methods. An automatic short answer assessment system based on NLP is attempted in this paper. Various experiments performed on a dataset, reveals that the semantic ENLP method out performs methods based on simple lexical matching; resulting is up to 85 percent with respect to the traditional vector-based similarity metric.

An automatic answer sheet checker checks the answer sheet and written mark as similar to human being. This software is built to check the subjective answer. The system consists of in build artificial sensor that verify answer and allocate marks according as good as human being accessing large number of handwritten answer sheet is relatively time-consuming task there is an intense need of speed up and enhance a process of rating hand written words while maintaining cost effectiveness.

II. PROPOSED SYSTEM

1. Dataset:

Keyword	Grammar	QST	Class
[1-6]	[0-1]	[1-6]	[1-9]

- Consists of 3 input labels namely:
  1. Keyword.
  2. Grammar.
  3. Question Specific Things (QST).
- And consists of a output label Class.
- Keywords, QST: (1-excellent, 2-verygood, 3-good, 4-ok, 5-poor, 6-verypoor).
- Grammar: (0-improper,1-proper).

2.Pre-processing:

- For doing this evaluation we have used 3 Parameters **namely Keyword, Grammar, QST.**

I. Keywords:

- Keyword is based on Cosine Similarity of "student's/user's answer" with "model answer".
- Firstly texts (i.e. student's and model answer) converted into vectors. And from these two vectors the Cosine similarity is Calculated.
- this gives value from 0 to 1. this is converted into numeric form (i.e. 0 to 100). And then keywords will get the value from 1-6

II. Grammar**:**

- For this number of grammatical mistakes taken into consideration.
- If no. of mistakes is less than threshold then it's given value 1 (proper) else it's given value 0 (improper).
- To find no. of grammatical mistakes we use API provided by https://textgears.com.

III. Question Specific Things**:**

Value of QST is obtained by measuring the Semantic similarity between the students answers and model answer

3: Scoring:

- Formula for calculating score for each answer:

Score = (predicted * out_of) / 10

- Above Score is the marks allotted for answer out of the value "out_of" which is chosen by admin.
- For example, predicted = 8, out_of = 5 then

Score = (8 * 5)/10 => 4 (out of 5).

III. IMPLEMENTATION

The project mainly divided into 4 modules, namely:

1. Semantic similarity between sentences
2. Text extraction
3. Keyword based similarity
4. Scoring

1. Semantic similarity:

The problem of calculating the semantic similarity between two concepts, words or sentences is a long-dealt problem in the area of natural language processing. In general, semantic similarity is a measure of conceptual distance between two objects, based on the correspondence of their meanings. Determination of semantic similarity in natural language processing has a wide range of applications. In internet related applications, the uses of semantic similarity include estimating relatedness between search engine queries and generating keywords for search engine advertising. In biomedical applications, semantic similarity has become a valuable tool for analysing the results in gene clustering, gene expression and disease gene prioritization. In addition to this, semantic similarity is also beneficial in information retrieval on web, text summarization and text categorization. Hence, such applications need to have a robust algorithm to estimate the semantic similarity which can be used across variety of domains.

Methodology:

The methodology considers the text as a sequence of words and deals with all the words in sentences separately according to their semantic and syntactic structure.

The information content of the word is related to the frequency of the meaning of the word in a lexical database or a corpus.

The method to calculate the semantic similarity between two sentences is divided into following parts:

- Word similarity
- Sentence similarity
- Word order similarity flow chart

Word Similarity:

We use word tokenizer and 'parts of speech tagging techniques implemented in natural language processing toolkit, NLTK. This step filters the input sentence and tags the words in to their 'part of speech'(POS) and labels them accordingly, to reduce the time and space complexity of the algorithm, we only consider nouns and verbs to calculate.

Example: 'A voyage is a long journey on a ship or in a spacecraft'

TABLE 1 Parts of speeches

Word	Part of Speech
A	DT - Determiner
voyage	NN - Noun
is	VBZ - Verb
a	DT - Determiner
long	JJ - Adjective
journey	NN - Noun
on	IN - Preposition
a	DT - Determiner
ship	NN - Noun
or	CC - Coordinating conjunction
in	IN - Prepostion
a	DT - Determiner
spacecraft	NN - Noun

Associating word with a sense

The primary structure of the WordNet is based on synonymy. Every word has some Synsets according to the meaning of the word in the context of a statement.

For example, word: 'bank.'

TABLE 2 Synsets and corresponding definitions from WordNet

Synset	Definition
Synset('river.n.01')	a large natural stream of water (larger than a creek)
Synset('bank.n.01')	sloping land (especially the slope beside a body of water)
Synset('bank.n.09')	a building in which the business of banking transacted
Synset('bank.n.06')	the funds held by a gambling house or the dealer in some gambling games

Consider an example where we calculate the shortest path distance between words 'river' and 'bank.' WordNet has only one synset for the word 'river'. We will calculate the path distance between synset of 'river' and three synsets of word 'bank'

Shorter distances for synset pairs are represented as follows

TABLE 3
Synsets and corresponding shortest path distances from WordNet

Synset Pair	Shortest Path Distance
Synset('river.n.01') - Synset('bank.n.01')	8
Synset('river.n.01') - Synset('bank.n.09')	10
Synset('river.n.01') - Synset('bank.n.06')	11

When comparing two sentences, we have many such word pairs which have multiple synsets. Therefore, if don't consider a proper synset right in the beginning it might lead to errors. Hence, sense of the word affects significantly on the overall similarity measure. Identifying sense of the word is part of the 'word sense disambiguation' research area. We use 'max similarity' algorithm, to perform word sense disambiguation.

$$\textbf{argmaxsynset(a)} = \sum_{i}^{n} \textbf{max(synset(i)(sim(i, a))}$$

Shortest path distance between synsets

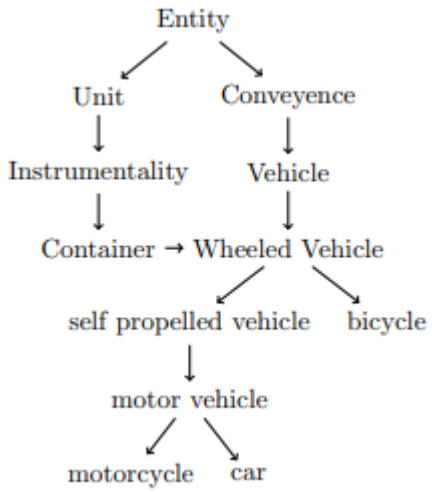

Fig. 1. Hierarchical structure from WordNet

consider words:

w1 = motorcycle and w2 = car
We are referring to Synset ('motorcycle.n.01') for 'motorcycle' and ('car.n.01') for 'car'.

The traversal path is: motorcycle → motor vehicle → car.

Hence, the shortest path distance between motorcycle and car is 2.

In WordNet, the gap between words increases as similarity decreases. We use the previously established monotonically decreasing function

$$f(l) = e^{-\alpha l}$$

Where l is the shortest path distance and α is a constant. The selection of exponential function is to ensure that the value of f(l) lies between 0 to 1.

Algorithm 1 Semantic similarity between sentences

```
procedure SENTENCE_SIMILARITY
    S1 - list of tagged tokens ← disambiguate
    S2 - list of tagged tokens ← disambiguate
    vector_length ← max(length(S1),length(S2))
    V1, V2 ← vector_length(null)
    V1, V2 ← vector_length(word_similarity(S1,S2))
    ζ=0
    while S1_list_of_tagged_tokens do
        if word_similarity_value > benchmark_similarity_value then
            C1 ← C1+1
    while S2_list_of_tagged_tokens do
        if word_similarity_value > benchmark_similarity_value then
            C2 ← C2+1
    ζ ← sum(C1, C2)/γ
    S ← ||V1||.||V2||
    if sum(C1, C2) = 0 then
        ζ ← vector_length/2
    Sim ←S/ζ
```

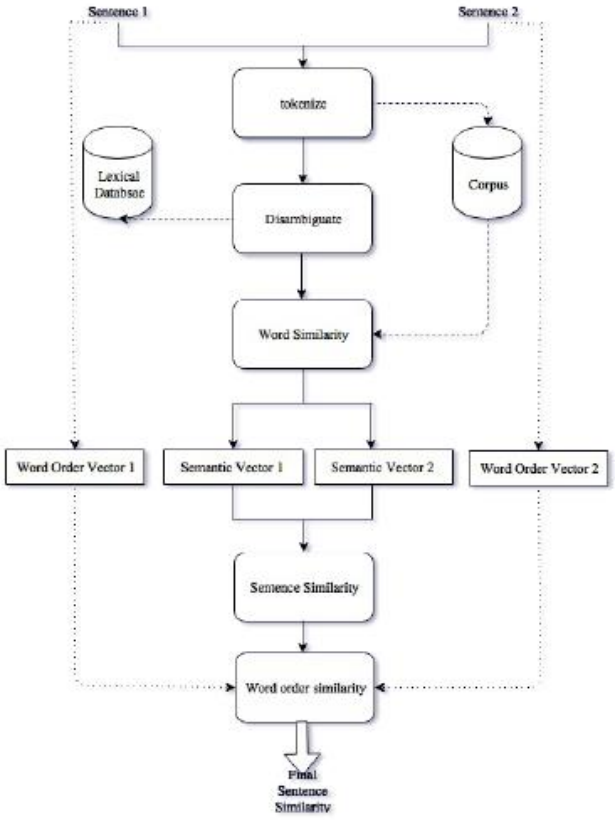

Fig. 2. Proposed sentence similarity methodology

2. Text extraction

Methodology:

In this process we extract the text from the scanned images of user answer sheet.

So, to improve the accuracy of OCR we perform the image pre-processing in 3 steps, namely: image rescaling, blurring, image binarization.

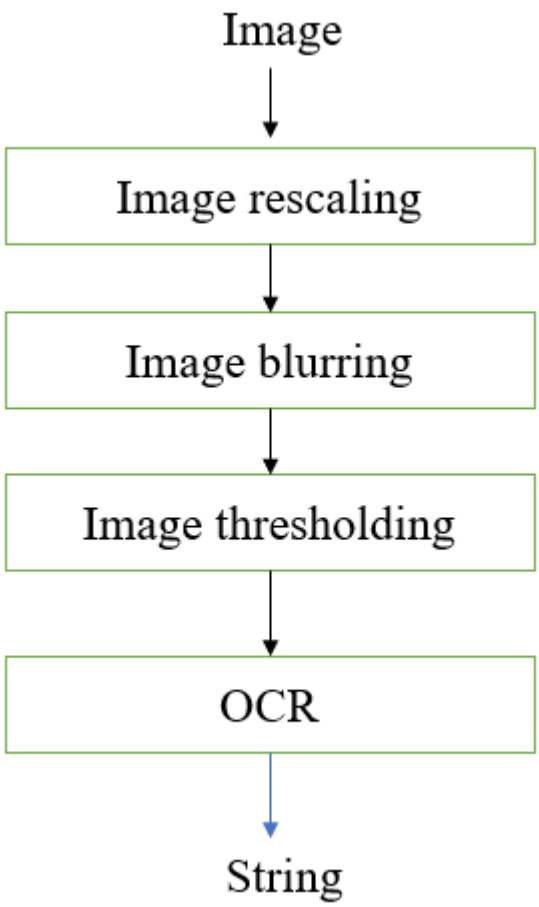

Fig. 3. Flowchart for Text Extraction

3. Keyword based similarity

Methodology:

Evaluation of keywords-based Cosine Similarity of "student's/user's answer" with "model answer".

Firstly texts (i.e. student's and model answer) converted into vectors. And from these two vectors the Cosine similarity is Calculated.

Now this gives value from 0 to 1. this is converted into numeric form (i.e. 0 to 100). And then keywords will get the value from1-6

Algorithm2 Keyword Based Similarity

procedure Keyword_Similarity:

1. convert text to lowercase
2. remove stop words
3. remove punctuation
4. reduce all word to its stem word
5. convert text to word vector
6. compute the cosine similarity for these two vectors
7. convert the cosine angle to keyword_score[1 - 6 : 1- ex, 6 - bad]

4.Scoring

Formula for calculating score for each answer:
Score = (predicted * out_of) / 10
Above Score is the marks allotted for answer out of the value "out_of" which is chosen by admin.
For example, predicted = 8, out_of = 5 then
Score = (8 * 5)/10 => 4 (out of 5).

Algorithm 3 Scoring

```
PROCEDURE check_answers:

text_extract_from_papers()
usr_ans_arr = get_firebase('user')
model_ans_arr = get_firebase('model')

count = 0
k, g, s=[]
p_value, marks = []
while count < len(usr_ans_arr) :
    k[count] = get_keyword_score(usr_ans_arr[count], model_ans_arr[count])
    mistakes = get_grammar_mistakes(usr_ans_arr[count])
    if mistakes <= t_value:
        g[count] = 1
    else:
        g[count] = 0
    s[count] = get_semantic_score(usr_ans_arr[count], model_ans_arr[count])
    p_value[count] = ml_model.predict(k[count], g[count], s[count])
    marks = (p_value * out_of)/10

upload_firebase(marks)
```

IV. CONCLUSION

Automated evaluation of short answers is done using algorithms the correct syntactically, semantically and assign scores. This reduces high cost and the slow turnaround of hand scoring thousands of written responses in standardized tests. We have to extract several features of the answers, ranging from simple numerical data such as word count and average word length, content specific numerical data such as miss spelt word count and adjective count and we look for specific

response in short answers. the system would assign preliminary grades to all answers, and the instructors would only become involved in the process to address student disputes. Automated grading if proven to match or exceed the reliability of human graders will significantly reduce costs. We have to implement and train machine learning algorithms to automatically access and grade answer responses. These grades from automatic grading system should match the human grades consistently. Finally, we need to analyze scores of sets of students.

V. FUTURE SCOPE

Third party tool can be used for the grammar checking and spell-checking module for better accuracy. With proper grammar checker the algorithm can also support compound and complex answers.

The spell checker module provides limited relaxation to the students. The available relaxation rules should vary according to different questions. There is provision to insert words in the files. New words can be added for new question-answer sets accordingly. The files can be upgraded so that the design can support a wide variety of answers.

REFERENCES

[1] D. Lin et al., "An information-theoretic definition of similarity." In Icml, vol. 98, no. 1998, 1998, pp. 296–304.
[2] A. Freitas, J. Oliveira, S. ORiain, E. Curry, and J. Pereira da Silva, "Querying linked data using semantic relatedness: a vocabulary independent approach," Natural Language Processing and Information Systems, pp. 40–51, 2011.
[3] P.W. Lord, R. D. Stevens, A. Brass, and C. A. Goble, "Investigating semantic similarity measures across the gene ontology: the relationship between sequence and annotation," Bioinformatics, vol. 19, no. 10, pp. 1275–1283, 2003.
[4] T. Pedersen, S. V. Pakhomov, S. Patwardhan, and C. G. Chute, "Measures of semantic similarity and relatedness in the biomedical domain," Journal of biomedical informatics, vol. 40, no. 3, pp.288–299, 2007.
[5] G. Varelas, E. Voutsakis, P. Raftopoulou, E. G. Petrakis, and E. E.Milios, "Semantic similarity methods in wordnet and their application to information retrieval on the web," in Proceedings of the7th annual ACM international workshop on Web information and data management. ACM, 2005, pp. 10–16.
[6] J. M. Sinclair, Looking up: An account of the COBUILD project in lexical computing and the development of the Collins COBUILD English language dictionary. Collins Elt, 1987.
[7] M. C. Lee, J. W. Chang, and T. C. Hsieh, "A grammar-based semantic similarity algorithm for natural language sentences," The Scientific World Journal, vol. 2014, 2014.