Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IMPACT FACTOR: 7.056

IJCSMC, Vol. 10, Issue. 10, October 2021, pg.33 – 39

DOMAIN SPECIFIC AUTOMATED ESSAY SCORING USING CLOUD BASED NLP API

George Pashev[1], Silvia Gaftandzhieva[1], Yuri Hopteriev[1]
[1]Computer Informatics Department, University of Plovdiv "Paisii Hilendarski", Bulgaria
georgepashev@gmail.com; sissiy88@uni-plovdiv.bg; yurih@uni-plovdiv.bg

DOI: 10.47760/ijcsmc.2021.v10i10.006

Abstract- The paper presents a methodology and an application framework (PUAnalyzeThis) that makes use of MeaningCloud API to automatically extract entities, concepts, relations, etc. and calculate scores and grades based on their relevance to a preliminary created topics graph. These topic graphs are either to be created by the teacher or automatically generated by scanning a certain amount of sample texts in the subject area. A prototype has been developed and tested with essays in the field of Computational Linguistics for informatics students at the University of Plovdiv "Paisii Hilendarski".

Keywords— automated scoring, framework, Computational Linguistics, students

I. INTRODUCTION

In light of the COVID19 pandemic, it has become very important for university professors to experiment with new approaches to evaluating essays. The fact that the teacher and the student often do not have eye contact in real time opens up great opportunities for fraud. The conditions in the various universities and faculties in Bulgaria are such that it is not always possible to provide real-time visual contact with means of synchronous communication such as Zoom. Not to mention that the very task of writing essays presupposes unattended work sometimes.

Even in the absence of pandemic conditions, it is sometimes necessary to assess the uniqueness of a text and the extent to which it relates to a predetermined subject area.

According to Burstein, Leacock and Swartz [1], essay questions are highlyvalued components of effective assessment programs, but the expense and logistics of scoring them reliably often present a barrier to their use. Zupan and Bosnić [2] also think that essays are the most useful tool to assess learning outcomes, guide students' learning process and measure their progress.One of the difficulties of grading essays stems from the perceived subjectivity of the grading process. Many researchers claim that the subjective nature of essay assessment leads to variation in grades awarded by different human assessors, which is perceived by students as a great source of unfairness [3].

Automated essay assessment is fast becoming a part of our digital teaching and learning world, and its possibilities have grown exponentially during recent years. Automated essay assessments a practical solution to a time-consuming activity of manual grading of students' essays. It reduces the costs. However, its main weakness is the predominant focus on the vocabulary and the text syntax. There is limited consideration of the text semantics [2, 4].Automated essay evaluation systems have been successfully used in large-scale writing assessments. However, existing systems mostly focus on grammar or shallow content measurements rather than higher-order traits such as ideas [5].

The automated essay evaluation has been the subject of some studies in recent decades. Many challenges have arisen in the field, including seeking ways to evaluate the semantic content, providing automated feedback, determining validity and reliability of grades and others [4].

The most common problems of the systems used in the early 21st century (e.g. Project Essay Grade (PEG), Intelligent Essay Assessor (IEA), Educational Testing Service, Electronic Essay Rater (E-Rater), C-Rater, BETSY, Intelligent Essay Marking System, SEAR, Paperless School free text Marking Engine, Automark, MY Access!) are the absence of both a good standard to calibrate human marks and a clear set of rules for selecting master texts, the lack of standard data collection, the need for a large corpus of sample text to train the system, the lack of feedback to students [3, 6, 7].

To conduct a more objective assessment of students in the discipline Computational Linguistics of the senior informatics majors studying full-time or part-time at the Faculty of Mathematics and Informatics, University of Plovdiv, we created and tested a prototype and a methodology, which we present in this paper.

The main problem with similar developments is that they use one well-known formalism to describe a subject area. Each of the formalisms for describing a subject area has its advantages and disadvantages and some aspects that it fails to describe. Therefore, as the first main goal (Goal 1), we identify the possibility of our model to support assessment by proximity to a subject area by using more than one formalism to describe a subject area.

In practice, it has been shown that each proximity coefficient calculated by invoking the corresponding algorithm in the different formalisms should be used with a different weight in the general formula for calculating the essay score. Therefore, we set as a goal (Goal 2) the ability to set a weighting factor for each of the algorithms used and the pre-selection of algorithms for calculating proximity to the subject area for each prototype.

In addition to determining the proximity of a text to a subject area, other variable characteristics of the text are important for the formula for calculating the overall grade. As a goal (Goal 3), we have set the possibility for our framework to support the setting of both static (built-in) variables and dynamic variables specific to each prototype built on the framework. Examples of static variables are the number of unique concepts, the number of unique entities, and the number of relationships between concepts in the text. Their weights have default values but can be changed for each prototype individually. Every developer who uses the framework must be able to quickly and easily set new variables, the formulas for their calculation, as well as the weights of both static and dynamic variables.

To prove that the framework is easy to use, we created a prototype for a topic in the syllabus of the discipline Computational Linguistics and using it automatically evaluated students' papers for this topic submitted in the last five years.

II. RELATED WORK

Burstein, Chodorow and Leacock [8] have developed a web-based system that provides automated scoring and evaluation of student essays. The system employs natural language processing and machine learning techniques to detect errors in grammar, identify discourse elements in the essay, recognize potentially undesirable elements of style and score the essay scoring system. All these capabilities outperform baseline algorithms, and some of the tools agree with human judges in their evaluations as often as two judges agree with each other.

McNamara et al. [9] compute essay scores using a hierarchical approach, analogous to an incremental algorithm for hierarchical classification. The features included in the analysis are computed using the automated tools, Coh-Metrix, the Writing Assessment Tool, and Linguistic Inquiry and Word Count. Overall, the models developed to score all the essays in the data set to report 55% exact accuracy and 92% adjacent accuracy between the predicted essay scores and the human scores. The results indicate that this is a promising approach to automated essay scoring that could provide more specific feedback to writers and may be relevant to other natural language computations, such as the scoring of short answers in comprehension or knowledge assessments.

Link [10] develops and validates an automated essay scoring engine to address the need for resources that provide precise descriptions of students' writing development. Development of the engine utilizes measures of complexity, accuracy, fluency, and functionality, which are guided by Complexity Theory and Systemic Functional Linguistics. The measures are built into computer algorithms by using a hybrid approach to natural language processing (NLP), which includes the statistical parsing of student texts and rule-based feature detection. Results from validation provide a mixed set of validity evidence both for and against the use of these measures for assessing development and underscore the possibilities of using computerized writing assessment for measuring, collecting, analyzing, and reporting data about learners and their contexts to understand and optimize learning and teaching.

Zupanc and Bosnić [4] comprise 21 state-of-the-art approaches for automated essay evaluation and highlight their weaknesses and open challenges in the field. They conclude that the field has developed to the point where the systems provide meaningful feedback on students' writing and represent a useful complement (not replacement) to human scoring. Systems recognize certain types of errors, including syntactic errors, provide global feedback on content and development, and offer automated feedback on correcting these errors.

A problem concerning the automated essay evaluation community is the unification of evaluation methodology. Later, Zupanc and Bosnić [2] propose an extension of existing automated essay evaluation systems by incorporating additional semantic coherence and consistency attributes. They design the novel coherence attributes by transforming sequential parts of an essay into the semantic space and measuring changes between them to estimate the text coherence. The novel consistency attributes detect semantic errors using information extraction and logic reasoning. The developed SAGE system provides semantic feedback for the writer and achieves significantly higher grading accuracy compared with other systems.

Crossley et al. [11] investigate a novel approach to automatically assessing essay quality that combines natural language processing approaches that assess text features with approaches that assess individual differences in writers such as demographic information, standardized test scores, and survey results. The results have important implications for writing educators because they reveal that essay scoring methods can benefit from the incorporation of features taken not only from the essay itself (e.g., features related to lexical and syntactic complexity) but also from the writer (e.g., vocabulary knowledge and writing attitudes).

Ramamurthy and Krishnamurthi [12] propose an automated answer evaluation system that uses cosine-based sentence similarity measures to evaluate the answers. They suggest 21 cosine-based sentence similarity measures and measured their performance using MSR paraphrase corpus and Li's benchmark datasets and use these measures for automatic answer evaluation system and compare their performances using the Kaggle short answer and essay dataset. The performance of the system-generated scores is compared with the human scores using Pearson correlation. The results show that system and human scores correlatewith each other.

Dong et al. [13] have built a hierarchical sentence-document model to represent essays, using the attention mechanism to automatically decide the relative weights of words and sentences. The model overcomes the weakness of the existing models built on recurrent neural networks and convolutional neural networks - different parts of the essay can contribute differently for scoring, which is not captured by existing models. Results show that the model outperforms the previous state-of-the-art methods, demonstrating the effectiveness of the attention mechanism.

Liang et al. [14] represent rating criteria by some sample essays that were provided by domain experts and defined a new input pair consisting of an essay and a sample essay. Researches propose a symmetrical neural network automated essay scoring model that can accept the input pair. The model termed Siamese Bidirectional Long Short-Term Memory Architecture (SBLSTMA) can capture not only the semantic features in the essay but also the rating criteria information behind the essays. The proposed model is used for the task of automated essay scoring and takes the Automated Student Assessment Prize dataset as evaluation. Experimental results show that this approach is better than the previous neural network methods.

Shankar and Ravibabu [15] have used features like a bag of words, sentence, and word count along with their average length, structure, and organization of an essay to achieve maximum accuracy in grading. They have also used a sequential forward feature selection algorithm to compare the accuracy and select the best subset of features. An efficient subset is formed from a single empty set because this algorithm and its operations made it easy to implement and perform well on small data sets.

Uto, Xie and Ueno [16] propose a hybrid method that integrates handcrafted essay-level features into a DNN-AES model. Their method concatenates handcrafted essay-level features to a distributed essay representation vector, which is obtained from an intermediate layer of a DNN-AES model. The method is a simple DNN-AES extension but significantly improves scoring accuracy.

Bhatt, Patel, Srivastava and Mago [17] suggest an automated essay scoring approach that involves not only rule-based grammar and consistency tests, but also the semantic similarity of sentences, thus giving priority to question prompts. Similarity vectors are used obtained after applying semantic algorithms and calculated statistical features. The system uses 22 features with high predicting power while considering every aspect a human grader may focus on. Predicting scores is achieved using the data provided by Kaggle's ASAP competition using Random Forest.

Yang et al. [18] propose a new formulation of graph-based features for concept maps using word embeddings to evaluate the quality of ideas for Chinese compositions. The concept map derived from the student's composition is composed of the concepts appearing in the essay and the co-occurrence relationship between the concepts by utilizing real compositions written by eighth-grade students from a large-scale assessment, the scoring accuracy of the AECC-I computer evaluation system. The results indicate that the proposed method deepens the construct-relevant coverage of automatic ideas evaluation in compositions and that it can provide constructive feedback for students.

III. ARHITECTURAL OVERVIEW

In order to make our everyday text analysis tasks easier to implement, we created a linguistic application framework PUAnalyzeThis. It's implemented in Java and Python programming languages. It is designed in a way it can be run as a parallel task for parallel Frameworks and Systems like Apache Hadoop. Tasks are described in our proprietary task description language, which is cross translated to a Python Task with the help of Python PLY library.

Fig. 1. shows the architecture of PUAmalyzeThis Framework.

Users communicate with the Framework Backend through WebApp or MobileApp for Android, which uses PUAnalyzeThis REST API implementation with Java ThreadPool based server. The REST API contains web methods for accessing and storing data for PUAnalyzeThis Tasks, Subject Area Desctiptions and Subject Area Algorithms. There are also web methods for invoking PUAnalyzeThis Tasks.

When a PUAnalyzeThis task is invoked, it is first read as a file from Hadoop File System (HDFS). Then it is cross translated to Hadoop Map Reduce Task in Python. The Generated Map Reduce Task is then invoked and through REST API Client it accesses functionalities from MeaningCloud API and Google Translate API if input text for PUAnalyzeThis task is different than English.

Reader can observe that our Goal 1 is architecturally achieved in PUAnalyzeThis. There are two distinct DB collections for Subject Area Algorithms and Subject Area Descriptions. A single generated task can invoke multiple Subject Area Algorithms which calculate proximity of Input Text to a given Subject Area Description, which has a structure adequate to current Subject Area Algorithm.

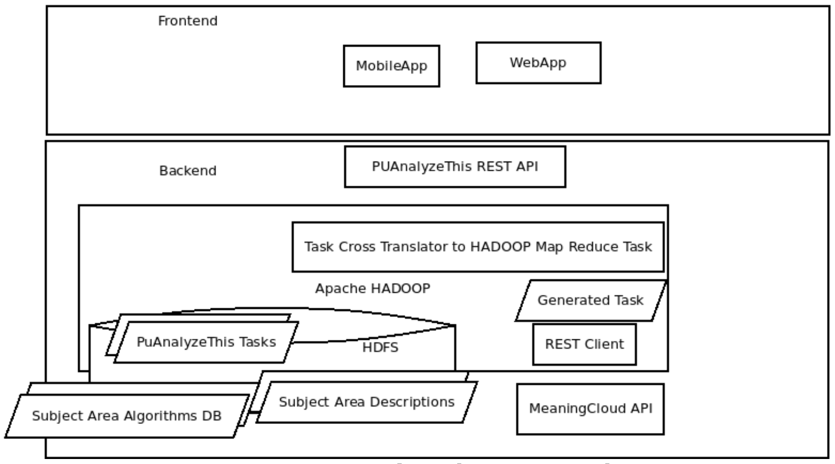


Figure 1. PUAnalyzeThis Framework

IV. PUANALYZETHIS TASKS

Our rationale to develop the framework was to make the description of Text Analysis Tasks easy and adequate for people, who do not necessarily have programming skills. That is why we paid much effort in creating a Task Cross Translator. In this section we discuss a sample PUAnalyzeThis Task.

```
Static Vars
nc = NumberOfConcepts, ne = NumberOfEntities,
nr = NumberOfRelations, nt = NumberOfTimeExpressions

Dynamic Vars
distConc = DistUniqConceptNums(Text, subjArea(CompLing))
distEnt = DistUniqEntities(Text, subjArea(CompLing))
compDist = (distConc - mindistConc)/(maxdistConc - mindistConc)
+ (distEnt - mindistEnt)/(maxdistEnt - mindistEnt)
Aggregates
maxNc = max(NumberOfConcepts), minNc=min(NumberOfConcepts),
# similar code ommitted ...

Grade Formula
0.3 * (nc - minNc)/(maxNc - minNc) + 0.2 * (ne - minNe)/
maxNe - minNe) + 0.5 * compDist
```

Figure 2. A Sample PUAnalyzeThis Task

Fig. 2. demonstrates a sample task. Typically a PuAnalyzeThis Task has several sections.

It is important to note, that in general each task is run for a collection (or dataset) of texts, rather than a single text.

The section of Static Vars introduces the static variables in the calculation. There are a bunch of built-in variables, which can be used in expressions in this section. For example, NumberOfConcepts contains the number of concepts discovered in the currently analyzed text in the collection of texts.

The section of Dynamic Vars introduces dynamic variables in calculation. Their expressions may contain functional expressions like DistUniqConceptNums which denotes invocation of corresponding Subject Area Algorithm for proximity calculation of Text to a given Subject area (for example subjArea(CompLing)).

The section of Aggregates introduces variables, which are calculated by using aggregate functions, which are invoked for the whole dataset (set of texts). Examples of standard built-in aggregate functions are max, min.

The Grade Formula section describes the formula, which is used by the framework to calculate the grade of each text in the dataset.

The reader can observe that Goal 2 & Goal 3 are achieved with this PUAnalyzeThis Task format. Static and dynamic variables capability is built-in in our proprietary task description language. Setting weighting factors is possible in the Grade Formula section and in any other section.

V. CALCULATION OF SUBJECT AREA RELEVANCE

A Subject Area can be described in many ways. Some approaches are discussed in Panayotova et al [19]. Here we give an example of description with Markov chain.

If we use concepts in the Subject Area as Markov Chain Graph Nodes, then any transition (vertices) from concept1 to concept2 is the probability for the appearance of concept2 after concept1 in any form of this word or any other synonym word in the text.

It can be easily observed that such a Markov Chain can be easily trained with large amounts of texts in this Subject Area. The more texts are used for training, the more accurate information is gathered in the Markov Chain.

The list *concepts* in Figure 3 contains the Markov Chain States, which are the concepts in the Subject Area. The list *vertices* contains labeled lists of vertices with calculated *p* for each vertex, which is the probability for the transition. The function *Calc_Distance* first invokes *stemSentence* which translates the text from Bulgarian language to English, then normalizes the sentence by removing backslashes and stems the words and then removes all words which are not in the concept dictionary.

```
  import numpy as np
  import random as rm
  #import the nltk package
  import nltk
  #call the nltk downloader
  nltk.download()
  from nltk.stem import PorterStemmer
  from nltk.stem import LancasterStemmer

  from nltk.tokenize import sent_tokenize, word_tokenize
  def stemSentence(text):
  token_words=word_tokenize(TranslateFrom(text, "Bulgarian"))
  token_words
  stem_sentence=[]
  for word in token_words:
  stem_sentence.append(porter.stem(word))
  stem_sentence.append("")
  return stem_sentence

  #code omitted ...

  concepts  =  [  {id:  1,  base_form:  concept1,  synset_id:  3  },  {...}...]
#concepts are preloaded from concepts database
  vertices = [{_from: 1, _to: 3, p: 0.3 }, ...]

  def Calc_Distance(text):
      text_stemmed = stemSentence(text)
      p = 0
      prev_word=""
      #p ends up with summed probabilities p for each vertice
```

```
        for base_word in text_stemmed:
              id1 = FindConceptID(base_word)
              id2 = -1
              if prev_word != "":
                    id2=FindConceptID(prev_word)
                    p = p + FindVerticeP(id2, id1)
              prev_word=base_word
        return p
```

Figure 3. A Python partial implementation of DistUniqConceptNums

The graph of the Markov Chain is Complete – there is a vertice for each possible transition, even vertices with p=0.

Then Calc_Distance sums up in p all p-s of all transitions for any sequence of concepti to conceptj where j=i+1 and returns the value of p. The bigger value p has, the more essential is this evaluated text to the Subject Area, according to our algorithm.

Similar to this algorithm DistUniqEntities is implemented. In DistUniqEntities entities are used instead of concepts as states in the Markov Chain.

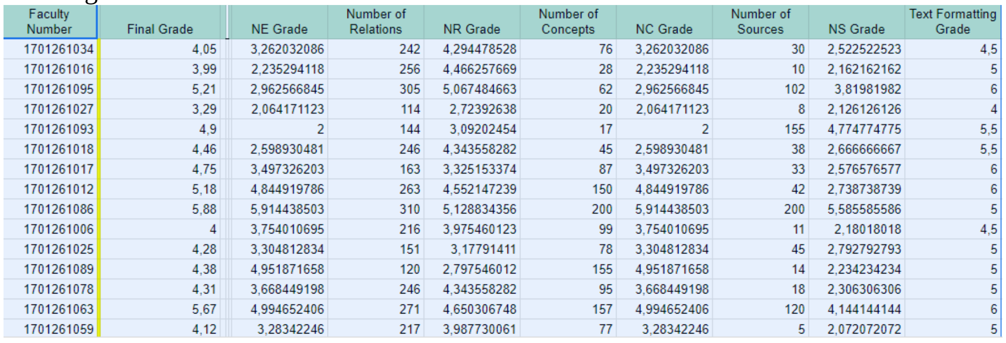
These algorithms are valuable in that they take into account not only the concepts and entities in the subject area, but also their sequence. This makes the assessment a little more accurate. They could also be used in combination with other algorithms that assess the relevance of a concept to a given text to achieve even more accurate assessment. As mentioned before, our approach supports usage of any number of such algorithms in a natural and straightforward way.

VI. EXPERIMENT AND RESULTS

An experiment was conducted for student essays in the discipline of Computational Linguistics for the specialty of Informatics at the Faculty of Mathematics and Informatics at the University of Plovdiv. Texts were taken, assessed as excellent by students from 5 years ago, and with them the respective Markov chains were created through machine learning. The list of concepts was set in advance by the teacher, and the probabilistic transitions were calculated according to the frequency of occurrence of the corresponding transition in the texts.

Subsequently, all student texts were evaluated against the Markov chains thus generated.

Then two teachers were given the texts and they had to re-evaluate them, not knowing what grades had been generated for them by this methodology. As a result, Teacher 1 reached 60% agreement with the automated procedure, and Teacher 2 72%. For the evaluation the standard scale in Bulgaria was used, from grade Weak 2 to grade Excellent 6.



Faculty Number	Final Grade	NE Grade	Number of Relations	NR Grade	Number of Concepts	NC Grade	Number of Sources	NS Grade	Text Formatting Grade
1701261034	4,05	3,262032086	242	4,294478528	76	3,262032086	30	2,522522523	4,5
1701261016	3,99	2,235294118	256	4,466257669	28	2,235294118	10	2,162162162	5
1701261095	5,21	2,962566845	305	5,067484663	62	2,962566845	102	3,81981982	6
1701261027	3,29	2,064171123	114	2,72392638	20	2,064171123	8	2,126126126	4
1701261093	4,9	2	144	3,09202454	17	2	155	4,774774775	5,5
1701261018	4,46	2,598930481	246	4,343558282	45	2,598930481	38	2,666666667	5,5
1701261017	4,75	3,497326203	163	3,325153374	87	3,497326203	33	2,576576577	6
1701261012	5,18	4,844919786	263	4,552147239	150	4,844919786	42	2,738738739	6
1701261086	5,88	5,914438503	310	5,128834356	200	5,914438503	200	5,585585586	5
1701261006	4	3,754010695	216	3,975460123	99	3,754010695	11	2,18018018	4,5
1701261025	4,28	3,304812834	151	3,17791411	78	3,304812834	45	2,792792793	5
1701261089	4,38	4,951871658	120	2,797546012	155	4,951871658	14	2,234234234	5
1701261078	4,31	3,668449198	246	4,343558282	95	3,668449198	18	2,306306306	5
1701261063	5,67	4,994652406	271	4,650306748	157	4,994652406	120	4,144144144	6
1701261059	4,12	3,28342246	217	3,987730061	77	3,28342246	5	2,072072072	5

Figure 4. Grade Calculation Results Fragment of Table

Figure 4 shows fragment of the automated calculation results for students with certain Faculty Numbers. The Final Grade in the format of the Bulgarian Grade system is shown. Some parameters and their partial grades are shown, which participate in the calculation of the Final Grade. Some of them are: Number of Relations, Number of Concepts, Number of Entities and number of References found in text by the Linguistic Analysis.

VII. CONCLUSION

The work demonstrates good achievement of the goals we have set in advance, as well as more than that. We have a framework that can be used quickly and easily in any case where an assessment of a collection of texts needs to be made. For future development, we will address the need for better parallelization of text evaluation algorithms, making better use of the fact that we use the Apache Hadoop cluster in the implementation architecture. An additional level of parallel work at the thread level in the nodes of the cluster could be used. It is also a good idea to provide an additional level of abstraction to ensure program independence as to which morphological analyzer is used and adopted to use of several well-known morphological analyzers, which support different sets of languages.

ACKNOWLEDGEMENT

The paper is supported within the National Scientific Program "Young scientists and Post-doctoral students" in accordance with Appendix No. 11 of Council of Ministers Decision No. 577 of 17 August 2018.

REFERENCES

[1]. J. Burstein, C. Leacock, R. Swartz, Automated evaluation of essays and short answers,*Loughborough University. Conference contribution*, 2001, https://hdl.handle.net/2134/1790

[2]. K. Zupanc, Z. Bosnić, Automated essay evaluation with semantic analysis, *Knowledge-Based Systems*, Vol. 120, 2017, pp. 118-132, https://doi.org/10.1016/j.knosys.2017.01.006.

[3]. S. Valenti, F. Neri, A. Cucchiarelli, An Overview of Current Research on Automated Essay Grading, *Journal of Information Technology Education*, Vol. 2, 2003, pp. 319-330.

[4]. K. Zupanc, Z. Bosnic, Advances in the Field of Automated Essay Evaluation, *Informatica,*Vol. 39, 2015, pp. 383–395.

[5]. L. Yang, T. Xin, F. Luo, S. Zhang, X. Tian, Automated evaluation of the quality of ideas in compositions based on concept maps, *Natural Language Engineering*, 2021, pp. 1-38, doi:10.1017/S1351324921000103

[6]. S. Dikli, Automated Essay Scoring, *Turkish Online Journal of Distance Education-TOJDE*, Vol. 7, No: 1, 2006, pp. 49-62.

[7]. S. Dikli, An Overview of Automated Scoring of Essays. *Journal of Technology, Learning, and Assessment*, Vol. 5, No: 1, 2006, pp. 4-35.

[8]. J. Burstein, M. Chodorow, C. Leacock, Automated Essay Evaluation: The Criterion Online Writing Service, *AI Magazine*, Vol. 25, No: 3, 2004, pp. 27-36.

[9]. D. McNamara, S. Crossley, R. Roscoe, L. Allen, J. Dai, A hierarchical classification approach toautomated essay scoring, *Assessing Writing*, Vol. 23, 2015, pp. 35-59.

[10]. S. Link, Development and validation of an automated essay scoring engine to assess students' development across program levels, PhD Thesis, Graduate Theses and Dissertations. 14567, 2015.

[11]. S. Crossley, L. Allen, E. Snow, D. McNamara, Incorporating learning characteristics into automatic essay scoring models: What individual differences and linguistic features tell us about writing quality,*Journal of Educational Data Mining*, Vol. 8, 2016, pp. 1-19.

[12]. M. Ramamurthy, I. Krishnamurthi, Design and Development of a Framework for an Automatic Answer Evaluation System Based on Similarity Measures, *Journal of Intelligent Systems*, Vol. 26, 2017, pp. 243-262, https://doi.org/10.1515/jisys-2015-0031.

[13]. F. Dong, Y. Zhang,J. Yang, Attention-based Recurrent Convolutional Neural Network for Automatic Essay Scoring, *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017),* 2017, Association for Computational Linguistics, pp. 153-162, 10.18653/v1/K17-1017

[14]. G. Liang, B. On, D. Jeong, H. Kim, G. Choi, Automated Essay Scoring: A Siamese Bidirectional LSTM Neural Network Architecture, *Symmetry, Vol. 10,* 2018, 682, https://doi.org/10.3390/sym10120682

[15]. R. Shankar, D. Ravibabu, Digital Report Grading Using NLP Feature Selection. In: Nayak J., Abraham A., Krishna B., Chandra Sekhar G., Das A. (eds) *Soft Computing in Data Analytics. Advances in Intelligent Systems and Computing*, 758, Springer, 2018, https://doi.org/10.1007/978-981-13-0514-6_59

[16]. M. Uto, Y. Xie, M. Ueno, Neural Automated Essay Scoring Incorporating Handcrafted Features, *Proceedings of the 28th International Conference on Computational Linguistics*, 2020, pp. 6077–6088.

[17]. R. Bhatt, M. Patel, G. Srivastava, V. Mago, A Graph Based Approach to Automate Essay Evaluation, *2020 IEEE International Conference on Systems, Man, and Cybernetics*, 2020, pp. 4379-4385, doi: 10.1109/SMC42975.2020.9282902.

[18]. L. Yang, T. Xin, F. Luo, S. Zhang, X. Tian, Automated evaluation of the quality of ideas in compositions based on concept maps, *Natural Language Engineering*, 2021, pp. 1-38. doi:10.1017/S1351324921000103

[19]. G. Panayotova, G. Dimitrov, P. Petrov, B. Os, Modeling and data processing of information systems. *In2016 Third International Conference on Artificial Intelligence and Pattern Recognition (AIPR)*, 2016, IEEE, pp. 1-5.