Proceedings of the Federated Conference on Computer Science and Information Systems pp. 355–358 DOI: 10.15439/2019F96 ISSN 2300-5963 ACSIS, Vol. 18

Automatic Assessment of Narrative Answers Using Information Retrieval Techniques

Liana Stanescu
University of Craiova, Faculty of Automation, Computers and Electronics, Craiova, Romania
Email: stanescu@software.ucv.ro

Beniamin Savu
University of Craiova, Faculty of Automation, Computers and Electronics, Craiova, Romania
Email: benisavu@gmail.com

***Abstract*—In this paper we propose a novel system for automatic assessment of narrative answers using information retrieval algorithms. It is designed to help professors to evaluate the answers that they receive from their students. It is a Java application that communicates through a REST API. This REST API has at its core the Lucene library and exposes all the great functionalities that Lucene has. The application has one UI for the students and one UI for the professor. The student will select the professor, select the question, upload the answer and send it. The professor will evaluate the student answer using the algorithms that will be discussed in this paper. Also in this paper a series of experiments will be presented, and their result will give us a better understanding of the algorithms and have a taste of how they work.**

I. Introduction

IN our days e-learning is becoming more and more popular because of the bene ts that it can o er. Because evaluating students on-line represent an important action, research in the domain has been focused on improving the online assessment systems, by integrating various methods for accomplishing this desire. The majority of online assessment systems have integrated numerous types of questions that can be evaluated and graded easily, based on direct matching: true/false, multiple choice, fill in the blank.

However, there is the general opinion that these types of objective tests are not enough. There are many topics, especially in the domain of human science, as well as technical domain, where the evaluation of a student cannot be complete without narrative answers. In this case, the student must formulize the answer to questions in the form of free text. Thus, it is desired for an online assessment system to integrate assessment of objective questions and narrative questions together for a complete evaluation of the students' capacity of assimilating information.

Nowadays, there is no viable method that can evaluate the answers given by the students. Having the computer understand our language and not only numbers will bring great benefit because they can processes faster than us and come up with solutions in seconds rather than hours, days, weeks or months. So we will give the computer a set of answers and compare it with the model answer which is the correct answer.

In our original work, the comparison between the student answer and the correct answer will be done by applying some information retrieval algorithms. Each algorithm will be taken one by one to investigate on how it works, how it compares to the other algorithms and what results it gives to the test data. The algorithms that will be used are Vector Space Model (VSM), Bigram and Language Analyser.

II. RELATED WORK

In the last years, there has been research in the domain of automatic assessment of narrative answers. The results have been integrated in certain academic or commercial platforms [4], [5], [6], [7], [8], [9], [10].

A classification of these techniques can be found in [4]:

- Statistical methods
- Text Categorization Techniques (TCT
- Information Retrieval algorithms.
- Full Natural Language Processing (NLP)
- Clustering
- Hybrid approaches that combine several techniques

Although the techniques may seem very different, the general idea that stands at the base of these systems is the same: to compare the student's answer (or candidate answer) with the teacher's ideal answer (or reference answer). The closer they are, the higher the student's score is.

There is a series of studies on the use of Information Retrieval Algorithms in automatic evaluation of free answers that indicates their efficiency in the domain. In [11] the author presents a comparative study of 5 algorithms. The model answers and student's answers are represented as vectors and then similarities between them computed by using cosine similarity. The obtained results are very satisfying. In [12] is also presented a comparison of 3 algorithms: Fingerprint, winnowing algorithms and the cosine similarity that are widely used to compare documents.

III. SYSTEM OVERVIEW

The system has a modular architecture [2] presented in the figure 1. The user interacts with the system through the Java Application which has access to the database and to the REST API. The application modules communicate through Data Transfer Object (DTO).

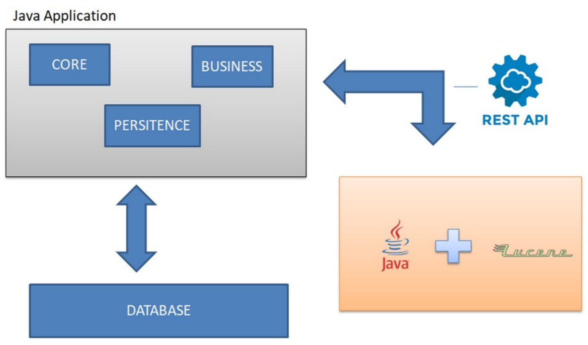

Fig. 1. Block Diagram of System Modules

A DTO is an object that carries data between processes. In other words, DTOs are simple objects that should not contain any business logic but may contain serialization and deserialization mechanisms for transferring data [3].

A. System Features

When the application is run the user will be prompted with a login screen and an option to register. The user will have to create a new account by selecting that option and fill up the register form.

In this system a user can have two roles. He can be a professor or a student. The application will provide to the student a combobox with the list of professors that are available. The student must select one of the professors. The selected professor will receive his answers. After this the student can choose whether to upload a file which contains the answer or insert the answer in the text area provided by the application. The student can edit his answer as long as the answer has not been send to the professor.

Once the answer is complete and the professor selected the student can now safely send the answer. The answer will be saved in the database and the professor that was selected will receive it for evaluation.

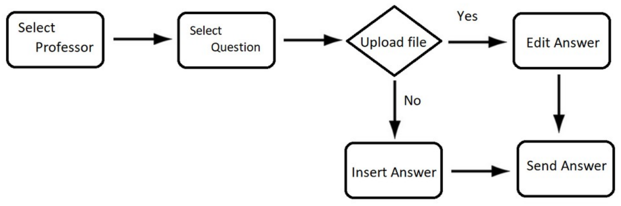

Fig. 2. Student activity diagram

The application will provide the professor with all the answers that the students submitted to him. The answers will come with the status of NOT_PROCESSED. The professor will now choose a reference text to evaluate the answers that were submitted. To evaluate the answers, the application will use different algorithms. But before the professor can evaluate, he must index all of the answers.

After the process of indexing is done the status for the answers will change from NOT_PROCESSED to INDEXED and he will be able to apply all the algorithms. The evaluation is done using the three algorithms: VSM, Bigram and Language Analyser. For each algorithm there will be a separate score. The higher the score, the closer it is to the correct answer entered by the professor.

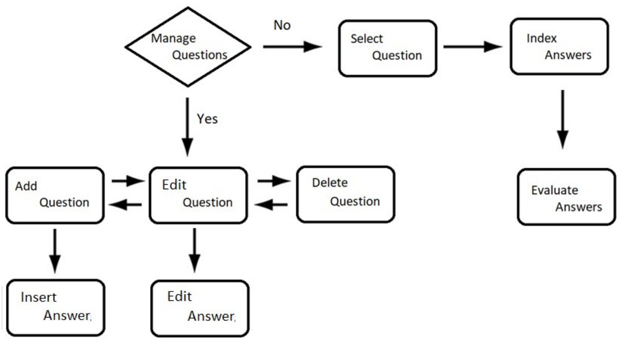

Fig. 3. Professor activity diagram

These are the high-level functionalities for the student and the professor, how they interact, what are the flows for them. Next we will take a closer look on how the index process works, how the algorithms work and all that is happening behind the scenes.

B. Inverted Index

Lucene uses a structure called an inverted index, which is designed to allow very fast full-text searches [13].

An inverted index consists of a list of all the unique words that appear in any document, and for each word, a list of the documents in which it appears.

To create an inverted index, we first split the content field of each document into separate words (which we call terms, or tokens), create a sorted list of all the unique terms, and then list in which document each term appears.

IV. ALGORITHMS

The algorithms are the heart of the system. Let's have a closer look at those algorithms and see how they actually work.

A. Vector Space Model

Once we have a list of matching documents, they need to be ranked by relevance. Not all documents will contain all the terms, and some terms are more important than others. The relevance score of the whole document depends (in part) on the weight of each query term that appears in that document. The weight of a term is determined by three factors [1]: term frequency, inverse document frequency and field-length norm.

B. Bigram

When words are used in conjunction with each other, they express an idea that is bigger or more meaningful than each word in isolation. The two clauses "I'm not happy I'm working" and "I'm happy I'm not working" contain the same words, in close proximity, but have quite different meanings [1]. If, instead of indexing each word independently, we were to index pairs of words, then we could retain more of the context in which the words were used. These word pairs (or bigrams) are known as shingles [1]. Of course, shingles are useful only if the user enters the query in the same order as in the original document. But this point is an important one: it is not enough to index just bigrams; we still need unigrams, but we can use matching bigrams as a signal to increase the relevance score [1].

Not only shingles are more flexible than phrase queries, but they perform better as well.

C. Language Analyser

Full-text search is a battle between precision, returning as few irrelevant documents as possible, and recall, returning as many relevant documents as possible. While matching only the exact words that the user has queried would be precise, it is not enough. We would miss out on many documents that the user would consider to be relevant. Instead, we need to spread the net wider, to also search for words that are not exactly the same as the original but are related [1]. There are several lines of attack: rRemove diacritics like ´, ^, and ¨ so that a search for rôle will also match role, and vice versa; remove the distinction between singular and plural; remove commonly used words or stop words to improve search performance; Including synonyms; check for misspellings or alternate spellings, or match on homophones—words that sound the same.

Lucene ships with a collection of language analyzers that provide good, basic, out-of-the-box support for many of the world's most common languages [13].

V. EXPERIMENTS AND RESULTS

We needed some datasets to test the algorithms, so for the first dataset we asked our friends to give 15 answers regarding any topic that they wants. They chose "sadness". The following text is the relevance answer: „Sadness is a normal feeling, an emotion we occasionally feel and should not be denied. Sadness is necessary; otherwise we could not appreciate the beautiful moments. It's normal not to feel good when you suffer a loss, when you're disappointed when something goes wrong."

A. Experiment 1

Answer: **Sadness is an emotion we** sometime are **feeling and should not** deny it. Without those **moments we** will **not be** able to **appreciate the** value of **the** world. This pain feels like a push of **the** soul, it is like **a** hollow in **the** soul that does not let you hope for better. **It's normal to feel** like this, to **be disappointed when something goes wrong**.

Score:VSM(18.4), Bigram(29.3), Language Analyzer(31.6)

We see that the answer contains quite a few words compared to the relevance text. Bigram scores very high because of the two word pairing that appear in the relevance text as well as in the answer text („Sadness is", „an emotion", „it's normal"). Language Analyzer algo gives us a high score because of the stemming function that it applies to the our texts, so words like "feeling", "feels" are transformed to "feel"," sadness" to "sad" and so on. Basically every word from the texts are stemmed to their root form, thus increasing the chances to find the same word multiple times.

B. Experiment 2

Answer: **Sadness is an** emotional pain we often try to hide from others. We experience **beautiful moments when we** are **not** sad. **It's normal not to feel good when you suffer a loss, when you're disappointed when something goes wrong**.

Score: VSM(26.5), Bigram(52.8), Language Analyzer(47.1)

We see that this answer contains an entire phrase that also appears in the relevance text and every algorithm gave us high scores. Here we see bigram scoring very big. By having the words in the exact order as in the relevance text, bigram scores higher, informing us that this answer is very relevant.

We will not show the results for the entire dataset. We chose only the ones that are the most interesting. The other results are similar with the ones presented above, either they are very close or they are very far or they are somewhere in the middle.

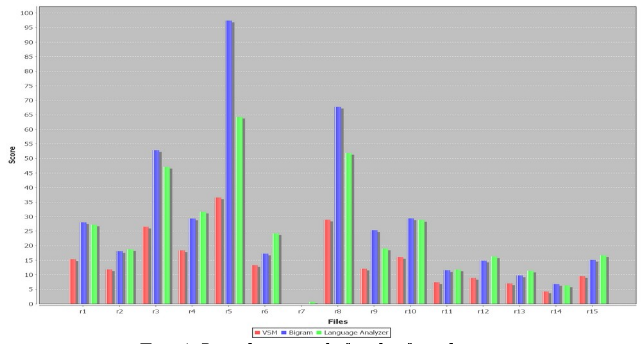

Fig. 4. Bar chart result for the first dataset

RED: VSM; BLUE: Bigram; GREEN: Language Analyser

As you can see in figure 4, for the first dataset, Bigram and the Language Analyser both perform better that VSM. This is because VSM takes into account only the words that are found in the answers, it does not perform any additional operations. Bigram searches through the answers not only with single word queries like VSM but also with pairs of 2 words, thus keeping some of the semantic for the respective answer. Language Analyser besides using the single word queries also uses the stemming of the words, so any variation of that word will be taken into consideration. Bigram and Language Analyzer are the way to go. Each one has their strengths and weaknesses. So on our dataset, for some answers bigram scores the most and for other answers Language Analyzer scores the most.

We will now look at the physics dataset. We will present the bar chart and the relevance order that the answers have for each algorithm (figure 5).

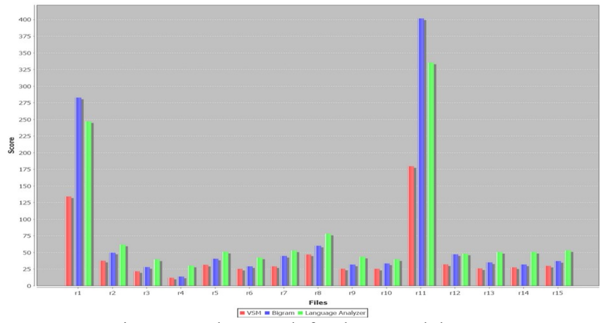

Fig. 5. Bar chart result for the second dataset

RED: VSM; BLUE: Bigram; GREEN: Language Analyser

First are VSM and Bigram. There is a big difference in the relevance order between them. By having a larger answer and a larger relevance text those results are normal for bigram, because bigram has more chances to find pairs of 2 words, thus making the answers more relevant.

Now let's look at Bigram and Language Analyzer. There are also differences in the relevance order between those two. Again this has to do with the fact that we have bigger answers and a bigger relevance text. So, the Language Analyzer is taking full advantage of its stemming operation, because a larger text means that the Language Analyzer will have more words that have the same root, thus a higher score.

The mean for each algorithm calculated from the data acquired in the 2 datasets appears in figure 6.

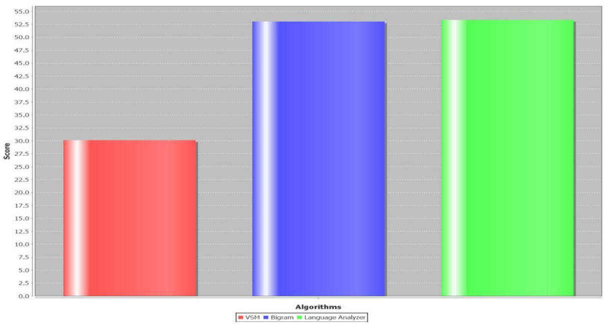

Fig. 6. Algorithms mean result

RED: VSM; BLUE: Bigram; GREEN: Language Analyser

We have the following results: VSM Mean: 30.1, Bigram Mean 53.05, Language Analyser Mean: 53.35.

As it can be seen the difference between Bigram and Language Analyser is very slim, advantage to Language Analyser. Either one of the two can be used to automatically evaluate answers.

VI. CONCLUSIONS

The paper presents our novel system that automatically evaluates narative answers using information retrieval techniques. As seen in the experiments above, Bigram and Language Analyser were the algorithms that performed the best, with Language Analyser having a small advantage over Bigram. But nonetheless both of them are very good and have scored good results on our test datasets.

Even though the objective was achieved it is not yet complete. Of course we can give scores to our answers, but we want some kind of mechanism to interpret this score and give that answer a grade. Also we might look to improve the algorithms to perform event better, a solution might be to combine Bigram and Language analyzer concepts. The performed experiments have shown satisfying results. For the time being, the module is integrated in an e-learning platform, in order to further be used and evaluated by the professors.

REFERENCES

[1] C. Gormley and Z. Tong, Elasticsearch: The Defenitive Guide available via web at https://www.elastic.co/guide/en/elasticsearch/guide/current/index.html

[2] G. Wielenga, "How to Split an Application into Modules?" https://dzone.com/articles/how-to-split-into-modules, 2009

[3] https://en.wikipedia.org/wiki/Data_transfer_object

[4] D. Perez, "Automatic evaluation of users' short essays by using statistical and shallow natural language processing techniques", Retrieved from https://pdfs.semanticscholar.org/025e/cb63d3322608e8f3073965ee9a0fc4d51e63.pdf, 2004

[5] D. Perez, E. Alfonseca and P. Rodrıguez, "Adapting the automatic assessment of free-text answers to the students profiles", Retrieved from https://dspace.lboro.ac.uk/dspace-jspui/bitstream/2134/2000/1/PerezD_AlfonsecaE.pdf, 2005

[6] D. Perez, O. Postolache, E. Alfonseca, D. Cristea, and P. Rodriguez, "About the effects of using Anaphora Resolution in assessing free-text student answer", in *Proceedings of Recent Advances in Natural Language Processing Conf.* Borovets, Bulgaria, pp.380-386, 2005.

[7] T. Mitchell, T. Russell, P. Broomhead and N. Aldridge, "Towards robust computerised marking of free-text responses", Retrieved from https://dspace.lboro.ac.uk/dspace-jspui/bitstream/2134/1884/1/Mitchell_t1.pdf, 2002

[8] J. Burstein, C. Leacock and R. Swartz, "Automated evaluation of essays and short answers", *in Proceedings of the International CAA Conference*. Loughborough: Loughborough University Davis, 2001

[9] E. Alfonseca and D. Perez, "Automatic assessment of short questions with a bleu-inspired algorithm and shallow nlp", *Advances in Natural Language Processing,* Springer Verlag, pp. 25–35, 2004

[10] D. Pérez-Marín, I. Pascual-Nieto and P. Rodríguez, "Computer-assisted assessment of free-text answers", *Knowledge Eng. Review*, vol. 24, no. 4, pp.353-374, 2009

[11] M. M. Hassan, "Experiments in Automatic Assessment Using Basic Information Retrieval Techniques", *Knowledge, Information and Creativity Support Systems*,Berlin, Springer, pp.13-21, 2011

[12] K. T. Tung, N. D. Hung and L. T. M. Hanh, "A Comparison of Algorithms used to measure the Similarity between two documents", *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET*), vol. 4, no. 4, pp. 1117-1121, 2015

[13] https://lucene.apache.org/