Short Texts Analysis for Teacher Assistance during Live Interactive Classroom Presentations

Michal Hucko*, Peter Gaspar*, Matus Pikuliak*, Vasileios Triglianos†, Cesare Pautasso†, Maria Bielikova*
*Slovak University of Technology in Bratislava, Faculty of Informatics and Information Technologies
name.surname@stuba.sk
†Software Institute, Faculty of Informatics, Universit della Svizzera italiana
name.surname@usi.ch

***Abstract*—We aim to improve the communication process of a teacher with students during lectures using question answering. Our work is focused on the analysis of students' answers to support the teacher in his lecturing. We work with students' answers to open questions, where it is impossible to identify finite number of solutions. In large classes it is impossible to react in real time to such answers since their evaluation is time consuming. We propose our own approach that helps the teacher by grouping similar answers. These groups are created based on proposed method employing text classification and clustering. Proposed method automatically estimates a number of clusters in answers using combination of k-Nearest Neighbors (KNN) algorithm and affinity propagation. We evaluated the method on real data in Slovak language collected from the course Principles of Software Engineering using real time presentation system ASQ.**

***Index Terms*—text clustering, text classification, technology enhanced learning, text preprocessing**

I. Introduction

An evaluation of the students' answers is a part of the teacher's work. During this process it is crucial to understand the answers and to be able to compare them in order to recognize problems in comprehension of the students. However, this is usually a time consuming task, since the teacher might potentially have hundreds of students. Checking the answers structured in a list is sub-optimal as we might find it difficult to uncover patterns or unusual behaviour.

Recently, an interactive approach to lectures gained a lot of attention. Some systems (such as ASQ[1] or Top Hat[2]) offer a combination of informative slides and slides where students are able to answer the questions or to solve some tasks regarding the explained topic. Here, a fast and reliable evaluation of students' answers becomes even a more important factor, hence if the teacher finds out what the problem is, he is able to change his explanations in order to improve student's experience during the lecture.

However, teachers need to understand answers coming in a stream in order to solve in real time misunderstandings in the class. Answers are usually displayed as an unstructured list ordered by the time stamp, or alphabetically, which also may cause trouble while checking. Thus, it can be time-consuming or even impossible when facing tenths and even hundreds of records. In real time it is also very hard to notice patterns while going through the answers one after another as they appear. These patterns can give the teacher an intuition about troubles with understanding of the class. Without this information the teacher can find it hard to change his explanations in order to help the students to better comprehend the content of the lecture. The additional explanations could support more students with troubles, which leads to better understanding of topic.

In our work, we propose a method for automatic structuring the students' answers. This process consists of two main stages. In the first stage, we detect the correct answers, which is provided by the text classification. The second step consists of clustering the incorrect answers to present them to the teacher in a well-arranged way.

We evaluate our approach using the real time presentation system ASQ that was used during the lectures of Principles of software engineering course at the Faculty of Informatics and Information Technologies Slovak University of Technology in Bratislava. After the domain experts annotate students' answers we compare these annotations to the method results using *adjusted rand index* [16]. We also use a method for predicting the number of clusters, which is capable of matching the experts estimations. Evaluation shows that the method has satisfying results on the collected data.

In the following section (Related work) we describe works on Slovak language preprocessing, text classification and clustering. We also mention methods for estimating number of clusters, which is important for the proposed solution. In section 3, we describe ASQ – a question answering system for live interactive presentations. Next, section 4 presents the proposed method for structuring students' text answers and section 5 evaluates the proposed method and discusses results on datasets gathered in Principles of software engineering course. The paper closes by its conclusions presenting summary and future work.

II. Related work

In our proposed approach, we focus on four techniques from the domain of the natural language processing: *text pre-processing*, *feature selection*, *text classification*, and *text clustering*.

[1] http://asq.inf.usi.ch
[2] https://tophat.com/

There is a lot of study done in the field of a text analysis. Usually the first phase is *text pre-processing*, which is particularly important for the Slovak language (since it is a flexive language). In a work of Gallay and Šimko [3] authors use lemmatization for the proper document preparation, but in many cases it is not sufficient. The time pressure of in-class exercises and quizzes can cause a lot of stylistic problems in the text of students' answers. Thus, also a grammatical processing is necessary.

According to a work of Wallach [15], one of the most common approaches in text processing is creating a *bag of words* model. It handles the documents as a set of independent words and does not take into consideration their order in a sentence. In a work of Papineni [8] a similarity between the result of online translators is measured. They want to determine whether a certain online translation is similar to a human one. For this purpose *an n-gram model* is used instead of the bag of words. N-gram is a sequence of n subsequent *elements* (Jurafsky [5]), in a case of text, these elements are words.

Algorithms that we use expect a matrix of numbers as an input (each row of such a matrix is a vector representing one document and columns are actual words). For this purpose a proper *feature extraction* with *feature selection* is required. In a work of Li and Zhang [7] different statistical approaches are compared for the feature selection. They conclude that in the field of text analysis a term frequency and inverse document frequency can be the reliable features.

For the term frequency, an equation for each feature i in the document j is as follows:

$$TF(i,j) = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{1}$$

where the total count of a term i in a document j (denoted as $n_{i,j}$) is divided by the total number of words in a document ($\sum_k n_{k,j}$). The modification of the term frequency *an inverse document frequency* (IDF) is giving the most frequent terms in all the documents a lower weight. It is described in a paper of Ramos [9]. This is notably useful for the words with high occurrence but little meaning (such as articles: *a* or *the*), which we also refer as *stop-words*. Since they can easily harm the results of text processing, to prevent this issue the following equation can be applied:

$$Tfidf(j) = Tf(j) * log(\frac{N}{Df(j)}) \tag{2}$$

where the original TF (term-frequency) term is multiplied by IDF (inverse document frequency) term. This represents the weight of given term j. In the equation N stands for the total number of terms in the corpus and the denominator df stands for the count of term j in corpus.

Inverse document frequency can be also utilized to handle n-grams. We can compute IDF for each gram in the corpora separately. Longer n-grams can be also made more important by adding a higher weight (Papineni [8]).

In the work of Tan [11] and Tong [14] for the *document classification* Support vector machine (SVM) and K-Nearest neighbor (KNN) are compared. It was revealed that SVM had the best results on a corpora with proper text processing. This includes namely lemmatization, tokenization and feature selection. KNN can be also used with the same configurations from SVM.

Next step is *text clustering*. Rangrej and Kulkarni [10] use various algorithms to cluster the tweets (short texts that are limited by the number of characters) gained from a social network Twitter[3]. From non-graph based algorithms they reported the best performance for the k-means algorithm. However, this approach requires to set the expected number of clusters as an input parameter. It is usually quite cumbersome to set this parameter without any previous knowledge about the corpus.

To deal with this problem, a technique called *Elbow function* is applied [6]. This approach is intended to minimize the cost function, which is a sum of squared distances between the document and cluster centroid. However, this method cannot be applied for every problem of text clustering, for example in cases when the elbow of cost function is not present. This problem may occur also in our data-set, because we cannot tell whether answers would have structure to fit the elbow. That means we have to predict the number of clusters.

III. ASQ

In our work we focus on a teaching process with the usage of real time presentation systems such as ASQ. According to the work of Triglianos [12], ASQ supports interactive presentations with embedded quizzes for technology enhanced lectures. Students can connect to them with their smart-phones or laptops. Once a teacher starts the session, the lecture slides are shared across all the connected devices.

Furthermore, slides containing open questions can be added. When such a slide appears, the teacher stops the explanation and let the students answer the question using their own devices. Meanwhile, the teacher can monitor the answering process using a separate teacher's view. This view lists the answers as they appear, so he/she is able to notice some common mistakes. This allows the teacher to adjust the lecture and to explain any problematic misconceptions.

Here, one problem may arise. In the current version of ASQ, the teacher is facing an unstructured list of students' answers. Even if the answers are sorted by their time stamp or alphabetically, having a large class with more than 100 students makes it challenging to monitor and manually deal with the large number of open-ended answers. Some common mistakes worth to comment on can be overlooked.

Currently, a real-time evaluation can be improved by proper preparation before a lecture. If the teacher sets up a finite a list of appropriated answers, ASQ provides a simple comparison of students' answers with this list and annotates the answers by OK or X sign. However, dealing with open text questions, setting up this list is very impractical, or even impossible.

[3]www.twitter.com

In our courses, we often pose short textual answer quizzes on the taught concepts. The student submissions often suffer from misspelling. Moreover, students have a limited time for answering. If we want to further analyze such answers, we need to take into consideration all these factors and try to deal with them.

Using supervised learning algorithms can help with automatic evaluation of answers for open text questions in ASQ. Clustering algorithms, on the other hand, can help us to better visualize results for these types of questions by grouping answers into groups by similarity. In the next section we describe method based on these approaches.

IV. Method for structuring text answers

The main goal of our work is to provide a valuable information about common mistakes of students to a person (mostly a teacher), who has to check the results efficiently and in real-time. Teachers would get structural view of the answers clustered in several groups. This method can be combined with real time class presentation systems, which provide methods to test the class through lecture.

Based on our analysis of students' behaviour in online learning systems, we identified several problems we need to deal with. As a consequence, we divided our method into the following steps:

1) *Answers preprocessing.* First, we need to gather data from the presentation system. This data (in a form of students' answers) is also categorized by the teacher into correct and incorrect ones. Here we may take into account an assumption that some questions tend to repeat from the past courses and can be reused in the following terms. Alternatively, an initial training set of pre-classified answers can be provided.
2) *Classification of answers.* Labeled answers are an input for the text classifier training. The text classifier is intended to predict the correctness of a new (non-labelled) answer of a student.
3) *Clustering of the answers.* Wrong answers obtained from the previous step are grouped into the clusters to help the teacher with fast uptake of the students' reactions to the lecture content. Since this step is unsupervised, number of clusters need to be estimated. For this purpose we use affinity propagation [2].

A. Answers preprocessing

First step of our method is a preprocessing of the answers. We employ existing approaches from the domain of natural language processing. The most common is lemmatization, stemming, and stop words removal. However, in case of Slovak language, these approaches are insufficient and other steps need to be applied such as spell check and grammar correction [2], [4].

People that are writing the text under the time pressure tend to make a lot of grammar mistakes. This scenario is very common especially at the lectures, where the students are asked to answer the teacher's questions. To prevent this issue, we check the grammar of the whole corpus. For this purpose, we use the spell checker implemented by Garenda [4], with the success rate of implementation is 98%.

For the lemmatization we use the *Slovak corpus lemmatizer*[4]. It is provided as an online service that converts the text into the list of corresponding lemmas. Lemma is the base word without any prefix or suffix which can be found in dictionary. Many languages, especially Slovak, have grammar based on this prefixes and suffixes. The success rate of Slovak corpus lemmatizer is around 67%.

The next step is a removal of punctuation. In our case, we just delete the punctuation. During this process we also change the capital letters to lowercase.

Finally, we modify the texts to reflect the fact that they are created in real time and often using mobile devices. We replace all the letters 'y' with 'i'. These two letters can be easily mistaken in Slovak language, since there is no difference in their pronunciation. The only way to use them properly is to memorize. Further preprocessing towards text correction would certainly improve the results depending on the particular dataset [1].

B. Classification of answers

The next step is the classification of students' answers. Before the application of the classification algorithms, a representation of answers as vectors of numbers is expected. For this purpose we use n-grams and feature selection techniques, which can support accuracy of classifier according to [5]. We experimentally set the maximum length of an n-gram to 4 and for features we compare two approaches: term frequency (TF) and inverse document frequency (TF-IDF).

We opt for SVM and KNN. For both of these algorithms a feature selection with TF-IDF model is appropriate. Since they are both supervised, we need a sample labeled data in order to train them. Here we may utilize students' answers that are categorized by the teacher into two classes: correct and incorrect.

Afterwards, the main task is to predict the correctness of the unknown answer. As we have mentioned, this approach can be applied, since the questions (for students) are reusable and they tend to repeat in some extent over the each instance of the learning course.

KNN marks the class of an unknown sample based on its K nearest neighbor in the training set (Tan, [11]). For measuring the distance between two points Euclidean distance or cosine distance can be used. In our cause we use Euclidean distance according to work [11], where authors got better results. SVM works on a little different principle, since it is trying to divide the sample space with the hyper line (Tong, [14]). That represents a border between samples from each class. One side of the line is the place where the samples belong to a certain class and another one is the opposite.

[4]http://text.fiit.stuba.sk/korpusovy_lematizator#focus.php

C. Clustering of the answers

For the clustering we use the k-means algorithm. Clustering with it belongs to the unsupervised learning problems, so we do not know anything about the true correctness of the answers. We can also use this method even when none of them are evaluated. In k-means k clusters are identified by the centroids. Each document belongs to one centroid depending on the shortest distance. There are many ways how to measure distances. In our work we apply Euclidean distance:

$$Dis(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (3)$$

For the second approach we use affinity propagation. It is a graph based method which is using a measure of document similarity for clustering. The algorithm divides the graph into suitable number of clusters. The closer two documents are, the more likely they will belong to the same cluster. Each vertex of the graph (document) communicates with others through messages. According to Frey [2], there are two types of messages. $Responsibility$ is sent from centroid i to its potential member j. Member is sending $availability$ to its potential centroid. Comparing it with k-means, the biggest advantage is in automatic estimation of k. In general, this approach gets worse results with texts. For the initialization we use Levenshtein distance.

Finally, we automatically estimate the number of clusters in k-means using affinity propagation as the first step. Results from affinity propagation are provided as an input to k-means.

V. Evaluation

Students used their smart-phones for answering questions posted during the class, which caused shorter answers. Also the questions were posted during the lecture where the lecturer maintained the flow of information, so the students had a very limited time for answering. Students used ASQ in the lectures of Principles of software engineering and their answers were written in Slovak language.

A. Dataset description

We performed two experiments, one for the classification task and the second for clustering task, as they require different properties of the dataset. For classification we used the dataset gathered from the course Principles of software engineering. It consisted of 28 open questions with 1 680 answers. These were labeled by teachers indicating their correctness. Students answered the questions in real time during the lecture in limited time using computers. They were motivated by receiving points for inputting the correct answers. For this dataset we labeled correct answers manually.

The purpose of the second experiment was to gather as much answers as possible at least for several questions to be able to estimate the number of clusters for incorrect answers. The second dataset was also gathered during the lectures of Principles of software engineering course using ASQ. Students were also motivated with extra points. For each lecture more than 100 active students were present. We had 8 such presentations during the whole course with more than 40 questions in total. Besides computers students were able to use their smart-phones to connect. Questions were the same as in the first dataset, however we do not have labels for correct answers. For the second experiment we chose only the questions with shorter answers (3 words in average). Dataset consists of totally 9 questions with more than 600 answers. Table 1 shows samples with the average answer lengths. Table 2 shows detailed view of each dataset composition.

TABLE I
Detailed view of dataset composition.

Purpose	Source	Labeled	Split	Size
Classification	Lectures no ASQ	Yes	Short answers	11 questions, 440 answers
			Long answers	17 questions, 680 answers
Clustering	Lectures ASQ	No	All answers	9 questions, 600 answers

TABLE II
Sample questions with average answer length.

Question	Average answer length (in words)
What are the basic features of feasibility study?	4
How is software quality defined?	4
What is the name of activity diagram element from picture. (Final node)	2
What is the name of activity diagram element from picture. (Fork, join)	2
What is the name of activity diagram element from picture. (Decision, merge)	2

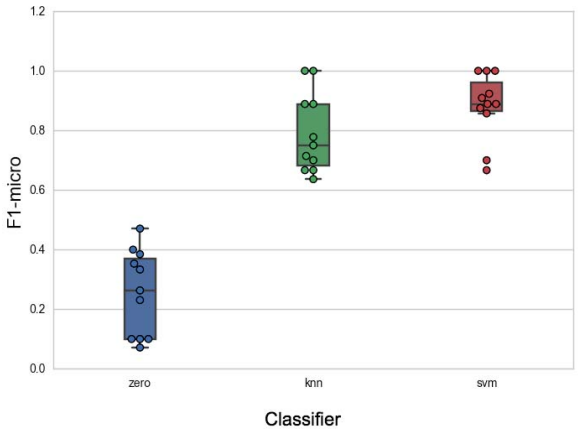

Fig. 1. Results for classification of short answer questions (with median length of answers less than 6 words).

B. Classification

We split the labeled dataset into two parts:

- first part were shorter answers, and

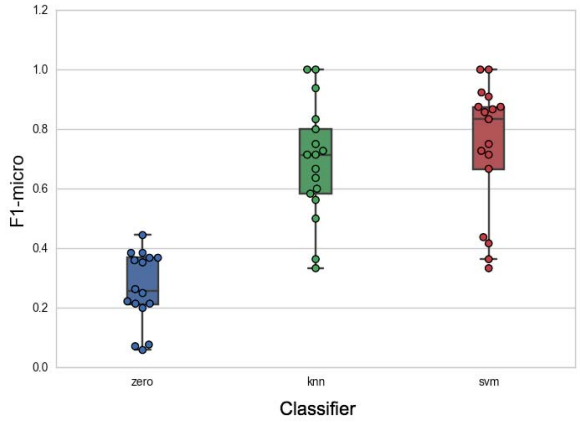

Fig. 2. Results for classification of longer answer questions.

- in the second part we left longer answers.

Reason for splitting the dataset was our expectation of different performance of the classifier for questions with different length. The decision on length of answers to be considered still as small was influenced also by number of answers in the dataset (median length less than 6 words).

We ran the SVM and KNN classifier on all questions (each question had more than 39 answers). For the implementation we used standard scikit-learn classes SGDCClassifier (SVM) and KNeighboursClassifier (KNN) with default values. We split each answer (for chosen questions separately) on training and test set (80% to 20%) and we computed F1-micro for each one. We chose micro, because of the structure of datasets when one class can be under-sampled. Finally, we compare the results with a zero classifier (always predicts the largest class for a given answer). Figure 1 shows the results. We can see that using SVM on short text classification is appropriate. Points in the graph are results for each question.

Figure 2 shows the results for longer answers (F1-micro measure). We can see that SVM has worse results here. It is difficult to train classifier on training sets with longer texts as they are indeed quite different even though they describe the same concept. Looking at charts we can see that the best results of F1-micro are around 85 % even when the train set is so small. This can support our scenario when the teacher does not have enough data from last years. Teachers can also train the classifier with their sample answers as we already mentioned in the section 4 (the preprocessing step).

C. Clustering

To see, whether the results are relevant for the teacher, we let three domain experts to cluster the second data-set containing more than 600 answers for 9 questions. We told them that the resulted clusters should reflect the common mistakes of the class. They were not told the number of clusters they should use. Table III shows the results of estimating cluster frequency. For this purpose we developed a simple visualization of answers. Our main concern was to make it effective for manual creation of clusters by experts.

We can see that our proposed approach (algorithm) has in most cases similar results as experts, except few cases when it predicts more clusters. If we look more closely, we can notice that estimations of domain experts are almost always different.

We computed an average adjusted rand index (ARI) through the experts. The metric compares their results in pairs (Figure 3). We can notice a spread which means that experts have problems to agree with each other.

TABLE III
NUMBER OF CLUSTERS CHOSEN BY EXPERTS AND ALGORITHM.

	q1	q2	q3	q4	q5	q6	q7	q8	q9
Expert 1	4	6	4	4	4	6	7	9	6
Expert 2	5	3	4	3	4	6	4	3	3
Expert 3	4	5	3	3	5	5	6	7	5
Algorithm	8	4	11	8	4	7	7	12	7

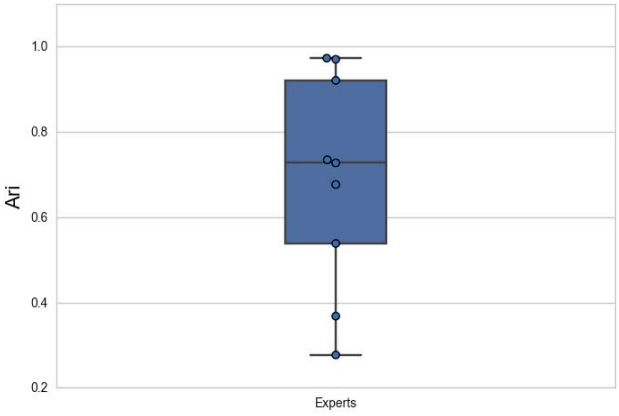

Fig. 3. Average adjusted rand index through experts.

Finally, we computed average an adjusted rand index (ARI) for each algorithm separately (Figure 4). 'Comb' stands for a combination of k-means 'km' and affinity propagation 'aff'. ARI was computed between pairs of expert and certain algorithm for each question. We also experimented with the methods with and without lemmatization. We can see that our method ('comb') achieved the best results. The results are also comparable with average ARI of experts, which means that clustering performed by human experts and our proposed methods were similar.

VI. CONCLUSION

Real-time presentation systems (such as ASQ) provide an interactive approach to lectures. In our work, we strove to improve teacher's experience by clustering the students' answers. Such clusters were meant to help the teacher better investigate how students understood lecture topic and help to find the common mistakes they made. We combined several algorithms in order to make this process more effective. We applied the text classification to automatically evaluate answers. Textual

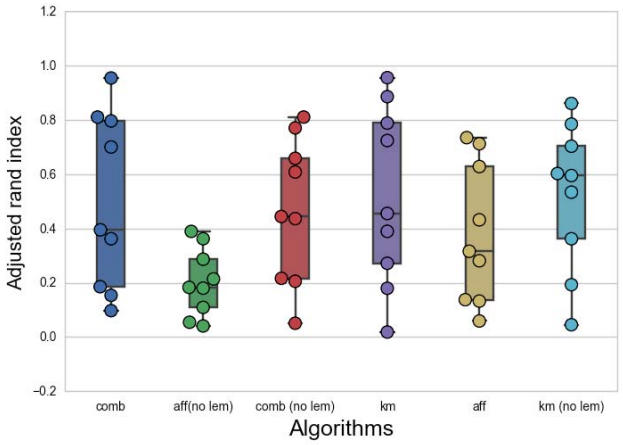

Fig. 4. Adjusted rand index for different clustering algorithms.

features were represented using the TF-IDF model. Clustering was used to effectively present incorrect answers in several clusters based on their similarity.

Our approach was evaluated using the two datasets: one for the text classification task and one for the evaluation of the clustering. Both of them consist of the same questions, but the first one was labeled by experts and has fewer answers. We revealed that the combination of affinity propagation and k-means algorithm generated results that are close to the domain expert estimations.

Based on lecturer's experience with ASQ we believe that our approach is able to enhance evaluation process of students' answers. In combination with a real time presentation system, it can further improve the lecture mainly the interaction between the lecturer and the students. The core part of our method (classification and clustering steps) is language independent. For extending to other languages we need appropriate tools for preprocessing step, i.e. lemmatization and grammar checking.

For the future work we may consider usage of this approach in lectures on learning programming languages where more precise answers can be gathered and so we can achieve even better results. Grouping similar source codes of students according to functionality may also help teachers in process of their evaluation and understanding problems in students' comprehension. We have already started the first experiments towards this direction [13].

ACKNOWLEDGEMENT

This work was supported by the Scientific Grant Agency of the Slovak Republic, grants No. VG 1/0646/15, VG 1/0667/18, the Cultural and Educational Grant Agency of the Slovak Republic, grant No. KEGA 028STU-4/2017 and it is the partial result of the collaboration within the SCOPES JRP/IP, No. 160480/2015.

REFERENCES

[1] Farzindar, A., Inkpen, D.: Natural Language Processing for Social Media. In Synthesis Lectures on Human Language Technologies Series. Hirst, G. (Series Editor). Second Edition. Morgan & Claypol, 2017.

[2] Frey, B. J.; Dueck, D.. Clustering by passing messages between data points. Science, 315.5814: 972–976 (2007)

[3] Gallay, L, Šimko, M.: Utilizing Vector Models for Automatic Text Lemmatization. In Proc. of SOFSEM 2016: Theory and Practice of Computer Science. Springer. To appear (2016)

[4] Gedera, J.: Diakritikovač slovenského textu. Supervisor: Dr. Marián Šimko. Bachelor's work. Faculty of informatics and information technology Slovak technical university in Bratislava, 35 (2015)

[5] Jurafsky, D., Martin, J.H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 189-233 (2000)

[6] Kodinariya, T. M., and Prashant R. M.. Review on determining number of Cluster in K-Means Clustering. IJARCSMS 1.6: 90–95 (2013)

[7] Li, T., Zhang, C., Ogihara, M.: A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. Bioinformatics, vol. 20, no. 15, 2429-2437 (2004)

[8] Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th annual meeting on association for computational linguistics, Association for Computational Linguistics, 311-318 (2002)

[9] Ramos, J., et al.: Using tf-idf to determine word relevance in document queries. In: Proceedings of the first instructional conference on machine learning, 1–4 (2003).

[10] Rangrej, A., Kulkarni, S., Tendulkar, A.V.: Comparative study of clustering techniques for short text documents. In: Proceedings of the 20th international conference companion on World wide web, ACM, pp. 111-112 (2011)

[11] Tan, S.: An effective refinement strategy for KNN text classifier. Expert Systems with Applications, vol. 30, no. 2, 290-298 (2006)

[12] Triglianos, V., Pautasso, C., Bozzon, A., Hauff, C.: Inferring Student Attention with ASQ. In: European Conference on Technology Enhanced Learning, Springer, 306-320 (2016)

[13] Triglianos, V., Labaj, M., Moro, R., Simko, J., Hucko, M., Tvarozek, J., Pautasso, C., Bielikova, M.: Experiences Using an Interactive Presentation Platform in a Functional and Logic Programming Course. In: Adjunct Proc. of UMAP '17, the 25th Conference on User Modeling, Adaptation and Personalization, ACM, 311-316 (2017).

[14] Tong, S.; Koller, D.. Support vector machine active learning with applications to text classification. Journal of machine learning research, 2. Nov: 45–66 (2001)

[15] Wallach, Hanna M. "Topic modeling: beyond bag-of-words." Proceedings of the 23rd international conference on Machine learning. ACM, (2006)

[16] Stenlei, Douglas.: Properties of the Hubert-Arable Adjusted Rand Index. Psychological methods, American Psychological Association, vol. 9, no. 3, 386 (2004)