International Journal of Management, Knowledge and Learning | ISSN 2232-5697

(How) Can Machine Learning Support Teaching Staff in a Virtual Collaborative Learning Environment?

Arne Böhmer
Technical University of Dresden, Germany
arne.boehmer@mailbox.tu-dresden.de

Maximilian Musch
Technical University of Dresden, Germany
maximilian.musch@mailbox.tu-dresden.de

Hannes Schubert
Technical University of Dresden, Germany
hannes.schubert1@mailbox.tu-dresden.de

Sebastian Schmidt
Technical University of Dresden, Germany
sebastian.schmidt@tu-dresden.de

Purpose: Supervising roles within virtual collaborative learning (VCL) environments face many different challenges and are often having difficulties keeping an eye on everything and comprehending every participant's or group's workflow. To help supervisors with problems like that we developed a software prototype to support their daily workflow and overcome the mentioned challenges.

Study design/methodology/approach: This study used a design science research approach to investigate how learning analytics might be able to support teaching staff in VCL settings. Previous studies demonstrated that this approach is very well suited to derive design guidelines for such software artifacts. Initially, a qualitative interview series with four experienced tutors was carried out to get an in-depth understanding of the challenges and tasks teaching staff typically faces in VCL environments. The interview material was systematically analysed to acquire the underlying software requirements. These were then combined with existing knowledge about support software of similar use cases to create a first prototype, primarily based on a supervised machine learning model that classifies online messages sent within the teams.

Findings: The software tool we have developed has shown that machine learning processes can indeed be used to support teaching staff in VCL environments. The tool achieves satisfactory classification performance in categorizing chat messages. It could therefore be demonstrated that it is possible to classify chat messages using a software tool. The interviews conducted revealed a particular interest in the statistical evaluations of group activities in Microsoft Teams. This was said to save a lot of time in the subsequent evaluation of the groups. The interviewed eTutors also expressed an interest in receiving private information on the development of possible conflicts within groups. The interest of the interviewed supervisors was very high and additional interviews could lead to further possible feature ideas. In general, the evaluative interviews resulted in positive feedback regarding how our prototype supported eTutors during their work.

Originality/value: In current times, spatially distributed and asynchronous education plays a bigger role than ever before. VCL environments are useful tools to overcome these challenges of time and space. However, unlike research on conversational agents or automated feedback systems, the focus of this work is not on the communication interface between software and users or the exclusively quantitative analysis of messages. The goal is to evaluate the content of sent messages within VCL environments using the six defined categories to gain a better qualitative understanding of the teams' collaboration and its workflow.

1 Introduction

In current times spatially distributed and asynchronous education plays a bigger role than ever before. Virtual collaborative learning (VCL) environments are useful tools to overcome challenges regarding the resulting distances between learners in time and space. In the context of this research VCL projects take place in settings where teaching staff (within our research scope, we were primarily focused on eTutors) assign tasks to the students, who then must organize their workflow amongst themselves to solve given problems. The eTutors are often responsible for multiple groups over longer periods of time which results in difficulties keeping an overview and providing every group with the necessary support and feedback (Lenk & Schmidt, 2022).

Another current big topic is Artificial Intelligence which has received greater importance for education in recent years since it brings the opportunity to support the teaching staff's workflow significantly (Salas-Pilco et al., 2022). The objective of this paper is the combination of both topics using a machine learning model that classifies incoming messages within VCL projects into one or many of six categories:

- On-topic (OnTo) – The students are talking about aspects concerning the topic of their tasks
- Organizational arrangements (Orga) – The students are organizing their workflow
- Discussion (Dis) – The students are discussing a certain topic
- Request for feedback (R4F) – The students are directly asking an eTutor for feedback
- Need for intervention (N4I) – The eTutor needs to step in
- Off-topic (OffTo) – No other category applies

To achieve that, we used a dataset consisting of roughly 1400 manually labeled messages sent within past VCL projects for training.

RQ1: Is it possible to train a machine learning model capable of classifying incoming messages within VCL projects using the given dataset that results in a sufficient classification performance?

Given RQ1 can be positively answered, the next goal is to find out how the eTutors workflow can be supported using the trained model. Therefore, interviews were conducted, and the software prototype was adjusted iteratively after reviewing their results.

RQ2: In what ways can a classification model for incoming messages within VCL projects be used to support the teaching staff's workflow and reduce their workload?

Since Artificial Intelligence in combination with learning analytics during VCL projects became a big topic in recent years, it is important to consider the existing related research (Salas-Pilco et al., 2022). To support the teaching staff's work there already are several concepts for conversational agents providing eTutors and students with statistics and information to improve formative assessment (Lenk & Schmidt, 2022). Furthermore, they also can be used to create automated feedback systems which notify students in case their contributions regarding the task lie below a certain limit (Porter & Grippa, 2020).

However, unlike research regarding conversational agents or automated feedback systems, this paper does not focus on the communication interface between software prototype and user or the exclusively quantitative analysis of messages. The goal is to evaluate the content of the sent messages with the help of the mentioned categories to get a better qualitative comprehension of the team's collaboration and their workflow.

2 Methods

In this study a design science research approach is applied, as this paradigm aims to create applicable, relevance-driven knowledge whilst also expanding the scientific knowledge base of the respective research branch (Hevner et al., 2004).

To get an in-depth understanding of the challenges that the teaching staff typically faces when hosting VCL projects, an initial series of qualitative, semi-structured interviews was conducted. During this step, four experienced eTutors with at least a year of teaching experience in international VCL environments were interviewed about the tasks they are usually dealing with and the areas of their work that might have potential to be supported by software. The interviews were recorded, transcribed, and systematically analyzed.

As proposed by Braun et al. (2015) the analysis of the interview transcripts focused on deriving explicit software requirements that a first prototype should fulfil to support the tutors according to their needs. For that reason, the context-specific needs of each tutor were identified and subsequently transformed into generalized software requirements. Furthermore, the existing knowledge base of scientific literature was reviewed to supplement the gathered requirements with pre-existing knowledge about software artifacts developed for similar use cases.

In the following phase a first prototype was designed that complies with the collected software requirements. As the developed software artifact is only intended to be used to validate the usefulness of the gathered software requirements and the subsequent software design, the prototype was implemented in an exploratory fashion (Pomberger & Blaschek, 1996). Such implementation enables the desired functionality to be conveyed with the help of application examples whilst keeping time for development short. The final implementation of the prototype was comprised of four basic components which are described by table 1 in more detail.

Table 1: Main components of the prototype

Component	Function	Technology used
Communication platform	Is used by teaching staff and students for communication via chats and calls. Relevant information is extracted via the built-in application programming interface (API).	Microsoft Teams
Database	Is used to persistently store messages and associated metadata.	MariaDB (based on MySQL)
Machine learning model	Is capable of classifying messages into categories.	Keras with a Tensorflow backend
Frontend and backend of the prototype	Utilizes the components to provide the desired functionality to the teaching staff in a graphical user interface.	Laravel with PHP & Blade frontend

In the final step the developed prototype was extensively evaluated to determine its usefulness and estimate its potential by acquiring first hand feedback from the teaching staff. The goal was to ultimately find out whether machine learning can facilitate support in this specific context of application and accordingly, which design principles of the prototype prove to be useful. For this purpose, a closing series of interviews with the same teaching staff was conducted. During these interviews the evaluation was conducted in two phases. In the first phase, the prototype was showcased with example data and the resulting impressions were collected using the thinking aloud method as described by Ericsson and Simon (1999). This method was selected since it is well suited to collect data about the usability of software applications and discover problems even with a relatively small number of users (Virzi et al., 1993). Accordingly, the interviewees were asked to verbalize their genuine, unfiltered thoughts during the demonstration of the prototype in the first phase of the interview series. In the second phase,

data about the utility of the prototype was collected by asking specific questions about the ways in which the teaching staff would use the prototype and which changes might need to be implemented to further improve its usefulness.

After analysing the data collected in the evaluative interviews, generalizable software design recommendations were derived and linked with knowledge from the existing scientific literature. The structure of the study's documentation was developed by following the suggestions by Gregor and Hevner (2013) to provide a meaningful contribution to the scientific knowledge base.

3 Results

In the following chapter we will present the results of our study following the design science research methodology by Hevner et al. (2004).

3.1 Problem identification

To gain a deeper understanding of the work of eTutors, we conducted four interviews with active and former eTutors. The focus was mainly on identifying challenges and daily tasks of an eTutor. Therefore, questions were asked about the usefulness of the training data's categories. The goal was to derive software requirements for our prototype from the conducted interviews. We identified three main challenges for eTutors in VCL environments:

- Time management: eTutors need to maintain constant awareness over usually three to four groups located in different time zones
- Evaluation: A weekly evaluation of each participant must always be done which can be a cumbersome and time consuming in VCL environments
- Traceability of students work: Up to now, the content of chat messages had to be searched out manually by the tutors in order to get a sense of the group's workflow.

Regarding the categories it has been mentioned several times that categories "Request for feedback" and "Need for intervention" are the most important ones.

3.2 Software requirements gathering

Based on the identified challenges we had to develop a concept for a software prototype that can support the challenges of eTutors. If we look at the research of Lenk and Schmidt (2022), we can see how important it is to present information to eTutors in a graphical and easily understandable manner. Based on this research and the challenges identified, we have established two main requirements:

- The prototype must be able to communicate a lot of information in a short time and in a comprehensible way.
- The prototype must be able to provide an objective evaluation of groups and members by eTutors. This should minimise the use of gut feelings should not play role in the evaluation.

3.3 Design process for the software artifact

Subsequently, a concept had to be developed to implement the determined requirements in the form of a software prototype. The trained machine learning classifier should be able to classify new chat messages into the trained categories. Based on a classification of the chat messages, which should be stored in a database, the goal was to generate comprehensive dashboards of the groups and the participants' activities. The dashboard should unite the identified requirements with the following design features:

- Message identifier: This should clearly show how many messages have been written and how many messages have been assigned to which category.
- Graphics: In general, the information should be presented in a comprehendible way.
- Drill down and drill up function: The dashboard shall be equipped with drill down and drill up functions to provide a high granularity of information. E.g., the entire group can be viewed, but also individual members.
- Time filter: Analyse activities within specific time periods.

3.4 Implementation of the software artifact

Implementation of previously described design was performed in two concurrent working steps, which were the development of a machine learning classifier on one side and on the other the development of a prototype which builds upon the functionality of that classifier.

3.4.1 Developing a message classifier

To develop a supervised machine learning model capable of classifying text messages into the six categories, inspiration was drawn from studies that have addressed or surveyed similar machine learning problems (i.e. Aggarwal & Zhai, 2012; Kowsari et al., 2019; Nowak et al., 2017).

During the development of a classifier, two approaches were pursued. While both were primarily based on a neuronal net, one disregarded the order of words in a message therefore relying on the Bag-of-Words assumption, whereas the other used the Long Short Term Memory (LSTM) originally introduced by Hochreiter and Schmidhuber (1997). For smaller datasets and vocabularies applying LSTM is more computationally expensive than similar architectures relying on the Bag-of-Words assumption, meaning that model training and optimization is challenging when computational power is relatively limited (Kowsari et al., 2019; Nowak et al., 2017). Nevertheless, in our context LSTM yielded significantly better results which is in line with the observations found in other studies (Nowak et al., 2017; Taspinar, 2016).

The following section will further detail our LSTM implementation since this was used in the final prototype. As mentioned in the introduction, our dataset involved 1412 messages distributed across six categories. As depicted in figure 1, the frequency distribution of labels within the dataset was skewed which meant that for some categories very few samples were available. Unfortunately, the categories occurring the least frequent were also the ones that the teaching staff showed most interest in during our interviews.

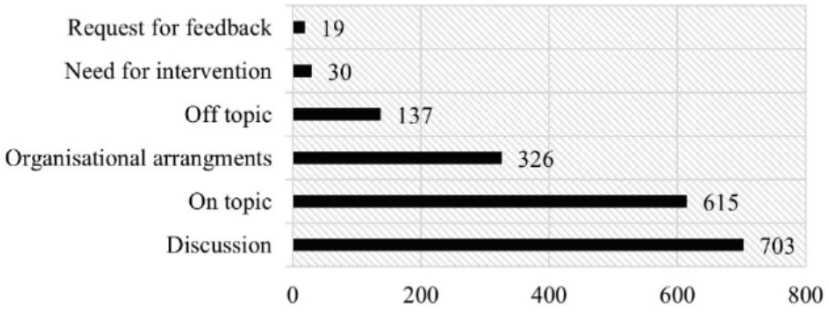

Figure 1: The frequency of each label in the 1412 messages of the dataset.

To optimally use the information existent in the dataset, extensive pre-processing was performed, and chat specific features were considered. During the text pre-processing, emojis, URLs and relevant punctuations were replaced with strings. Furthermore, contractions were expanded, and lemmatization of words was performed. To further optimize model performance, the GloVe word embeddings (Version 6B.100d) were used (Kowsari et al., 2019; Pennington et al., 2014).

After applying Bayesian hyperparameter optimization implemented by Nogueira (2014) to four different LSTM architectures and training them for up to 2000 epochs, the architecture illustrated in figure 2 could achieve the best results regarding F1-score performance. During training the data imbalance was dealt with by using a category-specific weighting. In the evaluation of the models, they were additionally tested at different thresholds for each category to determine the threshold combination that led to the best performance regarding recall and precision. Detailed information about the classification performance of the model used in our prototype can be found in table 2.

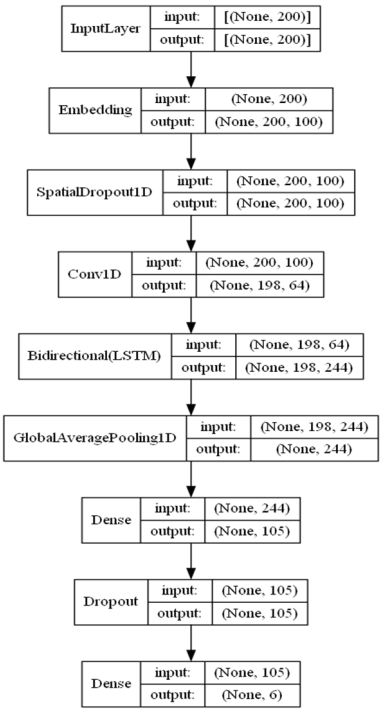

Figure 2: A visual representation of the model's architecture

Table 2: The performance of the model on a test dataset of 283 messages using category-specific thresholds

Category	Dis	OnTo	Orga	OffTo	N4I	R4F
Number of true samples in test dataset	215	181	125	25	7	4
Thresholds T_{F1} optimizing F1-Score	0.23	0.38	0.14	0.61	0.07	0.20
Per-category recall at T_{F1}	0.92	0.83	0.78	0.52	1	1
Per-category precision at T_{F1}	0.57	0.56	0.42	0.64	0.71	1
Per-category F1-score at T_{F1}	0.71	0.67	0.54	0.57	0.83	1
Macro-average F1-score at T_{F1}	**0.721**					
Hamming loss as defined in Tsoumakas and Katakis (2007)	**0.211**					

The model evaluation was based on recall, precision and F1-score performance as these metrics are better suited to evaluate model performance on imbalanced data than for example accuracy is. It should also be highlighted that ideally for each category a consideration should be made on whether better recall or precision would be preferrable if the classifier would be used in a live environment.

3.4.2 Using the classifier in a data extraction process

As mentioned in section 3.3 the goal is to create a dashboard to support the eTutors workflow as well as the assessment of groups and single students. Therefore, the created machine learning model must be embedded into a data acquisition process. During that process all relevant data is maintained in two tables: One for all messages and one for all users. After a message is sent by a student all the information (user ID, conversation ID, time stamp etc.) is extracted from Microsoft Teams with the help of an API call. Furthermore, the retrieved values are written into the two database tables. Once these updates are finished the model classifies the message content into one or many of the six given categories and writes the result in a new column of the messages table. The result is stored as binary data in the database, enabling further use within the dashboard. The whole database is connected to the frontend which is receiving live updates and provides a dashboard for the eTutors to get a better overview. The whole sequence is displayed in figure 3.

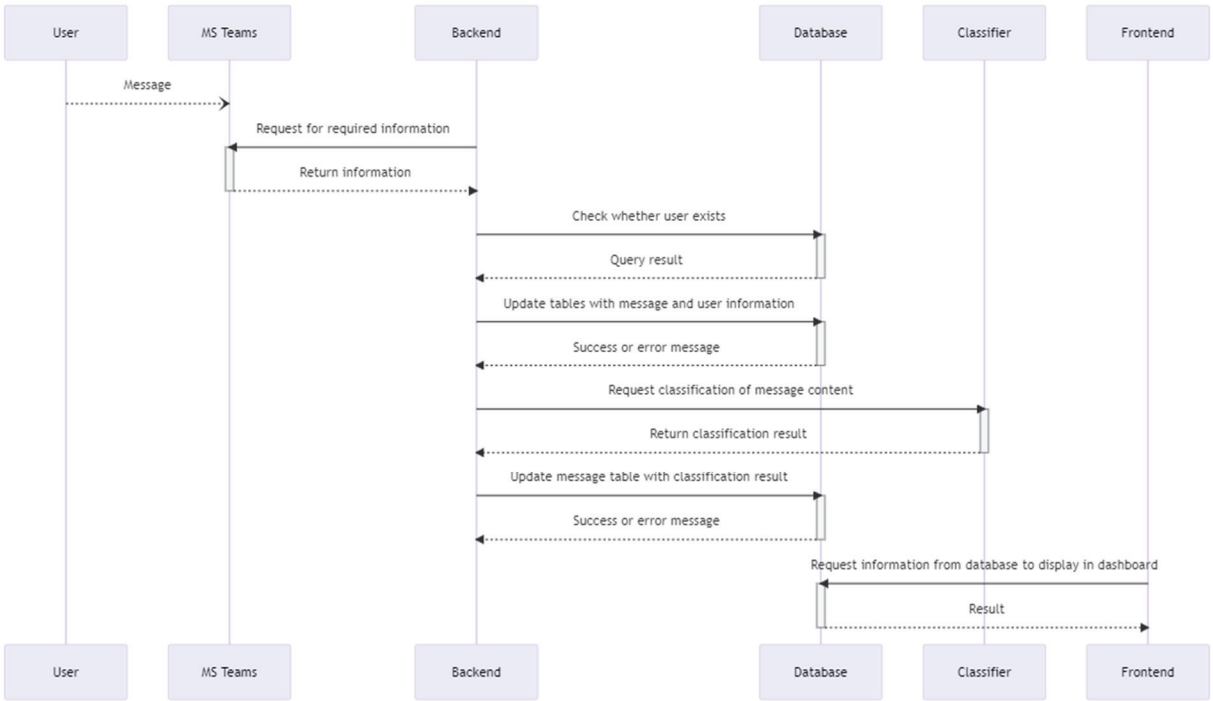

Figure 3: Data extraction process

The resulting dashboard fulfils all requirements mentioned in section 3.3. The eTutors can either view statistics regarding a whole conversation or for a single user. The visualized data has the purpose to support the assessment and feedback of teams and single students. Furthermore, a table with all messages and their according classification result can be viewed below a pie chart containing the distribution of messages within the selected scope. That way the data which is the basis of the graphical representation can be viewed and eTutors can get a better understanding of the models labelling which helps build their trust in the classification process.

3.5 Final interview phase

In general, the evaluation resulted in positive feedback regarding the support our prototype offered for eTutors. Three of the four respondents emphasized the possibility of a significantly reduced workload for eTutors. Thereby the supervision ratio, i.e. how many groups an eTutor can supervise, could increase significantly. In addition, the quality of supervision could increase as necessary interventions can be predicted in shorter time frames. This would give students the feeling of a better organized supervision. Most participants also requested a better comprehension of the given labels for each message so that their trust in the prototype's performance could be improved. Furthermore, our developed dashboard still received numerous suggestions for improvement through the formative evaluation, which are to be interpreted as future design features for the next iteration based on the prototype-artifact:

- Additional information about the length and content of the messages, role of the students and categories
- Possible comparison between selected groups or members which creates a better traceability
- Category ranking which shows which category occurs most often
- Providing a special dashboard version for students which summarizes the activity of their group and could promote self-reflection as well as the group morale

4 Discussion

In the context of this study, we created a classification model for incoming messages within VCL projects. In addition to that, we embedded this model into a data extraction process resulting in a support tool for eTutors. The goal was to find out whether we could train a model with a sufficient classification performance using the dataset we had available. Furthermore, we were looking for ways to support eTutors by giving the prototype the requested functionality.

RQ1: Using the given dataset the model achieved an F1-score of 0.721. This certainly is not good enough to deploy it in a production version yet but still resembles a promising result with potential improvement by future refinement (with more data and a modified, more consistent labelling process).

RQ2: As confirmed by the interviewed participants, the prototype offers functionality to support their daily workflow. Tasks that could be supported are: Getting a better overview of each group's collaboration and each students contributions, formative assessment of groups and students, getting an idea of which group needs feedback or an intervention in a short period of time.

Whether the utility of the prototype justifies its effort for implementation depends on various aspects. However, if previous work like this study can be used to build upon, machine learning

could provide a very valuable tool in the workflow of eTutors. Especially if the teaching staff is thereby enabled to supervise, monitor, and evaluate the students more efficiently, the quality and/or cost of teaching could be improved as a result. Furthermore, we argue that our study has significance even outside of VCL settings as communication in online environments is often at least partially based on chat messages and applying machine learning is such contexts should often be successful. Additionally, we want to stress that the solution we created in our study is only one way of utilizing the data generated by the machine learning model and other contexts might create many other use cases.

During the collection of software requirements, we moved within a narrow scientific scope as we conducted only four interviews with former and active eTutors from the Technical University of Dresden. Future research could extend this scope by interviewing VCL participants from other universities and by additionally including quantitative survey elements. This could yield further requirements and provide additional scientific evidence of the usefulness of our developed prototype. For the future we would consider several evaluation steps including additional interviews, which could lead up to a live test phase of our prototype. Overall, our research shows that chat analysis based on machine learning is a promising approach to support teaching staff in VCL environments.

References

Aggarwal, C. C., & Zhai, C. (2012). A Survey of Text Classification Algorithms. In *Mining Text Data* (pp. 163–222). Springer, Boston, MA. https://doi.org/10.1007/978-1-4614-3223-4_6

Braun, R., Benedict, M., Wendler, H., & Esswein, W. (2015). Proposal for Requirements Driven Design Science Research. In B. Donnellan, M. Helfert, J. Kenneally, D. VanderMeer, M. Rothenberger, & R. Winter (Eds.), *Lecture Notes in Computer Science. New Horizons in Design Science: Broadening the Research Agenda* (Vol. 9073, pp. 135–151). Springer International Publishing. https://doi.org/10.1007/978-3-319-18714-3_9

Ericsson, K. A., & Simon, H. A. (1999). *Protocol analysis: Verbal reports as data* (Rev. ed., 3. print). *A Bradford book*. The MIT Press.

Gregor, S., & Hevner, A. R. (2013). Positioning and Presenting Design Science Research for Maximum Impact. *MIS Quarterly*, *37*(2), 337–355. https://doi.org/10.25300/MISQ/2013/37.2.01

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *MIS Quarterly*, *28*(1), 75. https://doi.org/10.2307/25148625

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., & Brown, D. E. (2019). Text Classification Algorithms: A Survey. *Information*, *10*(4), 150. https://doi.org/10.3390/info10040150

Lenk, F., & Schmidt, S. (2022). The Virtual Tutor: Combining Conversational Agents with Learning Analytics to support Formative Assessment in Online Collaborative Learning. Advance online publication. https://doi.org/10.24251/HICSS.2022.009

Nogueira, F. (2014). *Bayesian Optimization: Open source constrained global optimization tool for Python*. https://github.com/fmfn/BayesianOptimization

Nowak, J., Taspinar, A., & Scherer, R. (2017). Lstm Recurrent Neural Networks for Short Text and Sentiment Classification. In (pp. 553–562). Springer, Cham. https://doi.org/10.1007/978-3-319-59060-8_50

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. http://www.aclweb.org/anthology/D14-1162

Pomberger, G., & Blaschek, G. (1996). *Software-Engineering: Prototyping und objektorientierte Software-Entwicklung* (2., überarb. Aufl.). Hanser.

Porter, B., & Grippa, F. (2020). *A Platform for AI-Enabled Real-Time Feedback to Promote Digital Collaboration*. https://www.mdpi.com/2071-1050/12/24/10243#cite

Salas-Pilco, S. Z., Xiao, K., & Hu, X. (2022). Artificial Intelligence and Learning Analytics in Teacher Education: A Systematic Review. *Education Sciences*, *12*(8), 569. https://doi.org/10.3390/educsci12080569

Taspinar, A. (2016). Sentiment analysis with bag-of-words. Retrieved from http://ataspinar.com/2016/01/21/sentiment-analysis-with-bag-of-words/

Tsoumakas, G., & Katakis, I. (2007). Multi-Label Classification. *International Journal of Data Warehousing and Mining*, *3*(3), 1–13. https://doi.org/10.4018/jdwm.2007070101

Virzi, R. A., Sorce, J. F., & Herbert, L. B. (1993). A Comparison of Three Usability Evaluation Methods: Heuristic, Think-Aloud, and Performance Testing. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *37*(4), 309–313. https://doi.org/10.1177/154193129303700412