Automated coding of student chats, a trans-topic and language approach

Adelson de Araujo*, Pantelis M. Papadopoulos, Susan McKenney, Ton de Jong

Department of Learning, Data Analytics and Technology, University of Twente, Postbox 217, 7500 AE, Enschede, the Netherlands

ARTICLE INFO

Keywords:
Automatic Content analysis
Collaborative learning
Machine learning
Multilingual language models
Transferability

ABSTRACT

Computer-Supported Collaborative Learning (CSCL) is known to be productive if well structured. In CSCL, students construct knowledge by performing learning tasks while communicating about their work. This communication is most often done through online written chats. Understanding what is happening in chats is important from both research and practical perspectives. From a research perspective, insight into chat content offers a window into student interaction and learning. From a more practical standpoint, insight into chat content can (potentially) be used to trigger supportive elements in CSCL environments (e.g., context-sensitive tips or conversational agents). The latter requires real-time, and therefore automated, analysis of the chats. Such an automated analysis is also helpful from the research perspective, since hand-coding of chats is a very time and labour-consuming activity. In this article, we propose a new machine learning-based system for automated coding of student chats, which we labelled ConSent. The core of ConSent is an algorithm that uses contextual information and sentence encoding to produce a reliable estimation of chat message content (i.e. code). To optimize usability, ConSent was designed in such a way that it can cover various topics and various languages. To evaluate our approach, we used two sets of chats coming from different topics (within the domain of physics) and different languages (Dutch and Portuguese). We tested different algorithm configurations, including two multilingual sentence encoders, to find the model that yields the best reliability. As a result, analysis revealed that ConSent models can perform with substantial reliability levels and are able to transfer reliable coding of chats in a similar topic and different language. Finally, we discuss how ConSent can form the basis for a conversational agent, we explain the limitations of our approach, and we indicate possible paths for future work to contribute towards reliable and transferable models.

1. Introduction

Data generated by digital educational platforms offer opportunities to build theories on learning and assess the effects of instructional interventions. In a process known as content analysis, for instance, researchers examine student conversations in Computer-Supported Collaborative Learning (CSCL) environments to create a body of scientifically valid evidence about an instructional intervention’s efficacy (Strijbos et al., 2006). To ensure the soundness of the conclusions drawn from content analysis, it is necessary to have a theory-based coding scheme (e.g., Weinberger & Fischer, 2006) and human resources to manually code all the samples. In the past two decades, several studies on CSCL have regarded text classification technologies as promising for automatically coding students’ discussions (Ferreira-Mello et al., 2019; Kovanovic et al., 2017; Rosé et al., 2008). In a modest scenario, these automated coding tools can reduce the role of the researcher to only check the coding outputs and make corrections where necessary. More ambitiously, such tools could function in real-time, e.g., in supporting the operation of conversational agents (Lalingkar et al., 2022; McLaren et al., 2010). But to date, successful cases of doing so in CSCL settings are limited.

Our research project is currently creating a conversational agent to aid students to engage in a productive learning dialogue as they participate in a collaborative group learning activity. Here, the agent’s use case is to prompt students in the chat to engage conversation that is conducive to student learning, given a pedagogically grounded intervention strategy (e.g., Michaels & O’Connor, 2015), e.g. by giving examples, elaborating ideas, expanding explanations, agreeing or disagreeing. Technology-wise, many commercial solutions are built to support single-user interaction, but group conversations have very different dynamics, so an alternative design must be employed. The biggest design challenge here is deciding what to say, to whom, and when, as explained by Kumar and Rosé (2010). Additionally, we also aim to design the conversational agent so that teachers can quickly

* Corresponding author.
E-mail address: a.dearaujo@utwente.nl (A. de Araujo).

https://doi.org/10.1016/j.caai.2023.100123
Received 4 October 2022; Received in revised form 9 January 2023; Accepted 9 January 2023
Available online 10 January 2023
2666-920X/© 2023 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).
configure it to function in various learning topics and for students with different characteristics. To handle these challenges, we argue that the conversational agent engine has to measure the presence of theory-based categories in chats using a data-driven mechanism to decide in real time on meaningful interventions. Therefore, a reliable artificial intelligence (AI) tool to perform automated content analysis on student chats is necessary. Such an automated coding mechanism can additionally be used in the development of AI-based learning systems, for example, to support teachers in making instructional decisions (Chiu et al., 2023) or feeding learning analytics dashboards (Chen et al., 2015).

In the current paper, we propose ConSent, a novel machine learning algorithm to automate content analysis of students’ chats. As we will describe, it uses Contextual information and Sentence encoding as core mechanisms. Its purpose is to estimate the sequential codes that reliably match those provided by humans, such that the conversational agent engine that relies on it can identify fruitful moments for intervening. Given a chat message, ConSent estimates the probability of each code given the previous codes (i.e., contextual information) and the input text (via sentence encoding). The coding scheme used in this paper was designed by (Eshuis et al., 2019) who used this scheme to categorize student chats into communicative intents based on the theoretical work by Saab et al. (2007). In the following, we outline the theoretical foundations for automated content analysis of collaborative learning processes in light of the recent advances in natural language processing and machine learning. Then, we present our research questions.

1.1. Background

Early CSCL research mostly relied on quantitative analysis, using surface-level metrics that could possibly indicate the quality of communication (Strijbos et al., 2006). Some of these metrics are, for example, the number of messages sent, the average number of words in a message, and linguistic metrics derived from dictionary-based approaches (e.g., Pennebaker et al. (2001)). These measures were later regarded as insufficient to describe learning outside of very narrowly constrained contexts (Rosé et al., 2008; Strijbos et al., 2006). As a consequence, studies started to rely on content analysis, i.e., the process in which “communication is coded, summarized, and frequencies/percentages are used for comparisons and/or statistical tests” (Strijbos et al., 2006, p. 30). Coding of student dialogues is mostly still done manually. However, manual coding is (too) time-consuming. Therefore, there is a high motivation to reduce this burden on human resources for coding and to design a coding tool that reaches high reliability (congruence with human coding).

Studies involving automated content analysis of collaborative learning data have been conducted based on a variety of theories (coding schemes), data channels (e.g., chats, forum posts, and transcribed video or audio), and units of analysis (e.g., message, sentence, thematic unit). These studies usually propose machine learning algorithms to process manually coded training samples and evaluate them with unseen test samples. For example, drawing upon the coding scheme of argumentative knowledge construction from Weinerberger and Fischer (2006), Rose et al. (2008) proposed a classification model using a feature-based machine learning algorithm. Feature-based machine learning algorithms have been employed to automate content analysis of a variety of instructional theories. Kovanci et al. (2016) focused on the “presences” in the community of inquiry framework proposed by Garrison et al. (1999) and Osakwe et al. (2022) recently focused on the feedback types proposed by Hattie and Timperley (2007). The work by Rose et al. (2008) also opened a broader discussion about the motivation, methodological challenges, and possible applications of automated content analysis in CSCL research. Rose et al. (2008), for example already suggested the potential for “triggering context sensitive collaborative learning support on an as-needed basis” (p. 237) which (in the form of a conversational agent) is also the ultimate goal of our work.

When designing data-driven tools for learning, a frequent concern is if, and how well, the technology aligns with a pedagogical theoretical framework (Jivet et al., 2017). In our specific use case, the conversational agent is meant to encourage groups of students to explicate their thinking. Specifically, the prompts in the chat are designed to stimulate students to elaborate and explain their observations and reasoning in relation to what has been stated already. To decide on what the agent should say to the students, we follow previous studies that used the Academically Productive Talk (APT) framework from Michaels and O’Connor (2015). For example, the agent can ask, “Bart, could you elaborate on what Milhouse just said?” or “Could someone recap everything that has been covered so far?” However, to decide on when to intervene, current conversational agents designed with simple vocabulary-based rules are unable to sufficiently draw on conversational categories when deciding on triggering conditions since they can only identify when the students refer to the topic keywords. To address this gap, conversational agents in other domains of application are increasingly relying on the message classification (of intent) as one of the vital sources for effective natural language understanding (Zhang et al., 2020). In the context of a collaborative group learning activity, this classification enables the conversational agent to automatically filter chat messages of interest based on content analysis. Thus, ConSent aims to provide conversational agents with a solid, theory-based, chat analysis that can guide interventions.

The major challenge is to create a tool that renders codes that resemble those of a human coder. The core of our coding system uses machine learning techniques and focuses on modeling chat messages given their text and context. We will give more technical details later in Section 2. A potential obstacle is that the text is often very short, full of internet slang, and thus highly context dependent. For example, a message “ok” can be coded in different ways, depending on what comes earlier. For this issue, using contextual information is a potential solution. The study from Shibata et al. (2017, pp. 65-71) focused on automated content analysis of chat data in Japanese, proposing their own coding scheme with 16 codes. By using context information, i.e., adjacent samples in the chat, they proposed a reliable Seq2Seq deep learning algorithm that outperformed feature-based baselines and other deep learning algorithms that did not account for context information. Besides suggesting that future work should work with coding schemes that are more sensitive to collaborating processes, these authors recommended subsequent algorithms consider multiple preceding contributions as contextual information (Shibata et al., 2017, p. 70). Yet, it is unclear how many preceding contributions should be employed by the algorithm as contextual information to complement what is given in the text input.

In addition to enabling reliable coding that can form the basis for a conversational agent, our model has two additional challenges. The first additional challenge is making a tool that works in topics not included in the training dataset. Ideally, a conversational agent should be useable on different topics. At the same time, it is not feasible to train a separate model for every topic. In other words, the model needs to be transferable to other topics. This transferability of a model is usually not included in existing work (see Fiacco & Rosé, 2018). A second major additional challenge is to make the model transferable to other languages. This will allow the possibility of training a model in one language and using it to code samples in other languages. With such a feature, teachers and students around the globe could use the same conversational agent. In addition to these practical advantages, we argue that transferability to different topics and languages can assist CSCL researchers in efficiently analyzing a more diverse range of learning environments. These challenges mentioned should be tackled in the design of the system.

As we consider transferability to other topics and languages a crucial usability requirement for our conversational agent, our machine learning algorithmic choices should also reflect these needs. The feature-based approach generally consists of manually selecting features (variables) that exploit context for increasing classification performance.
Some examples of relevant features include the presence or not of punctuation and rare words, text length, frequency of unigrams and bigrams, named entity recognition, and others. However, models trained with the feature-based approach are hard to transfer to datasets in another topic or language. The reason is that the resulting vector space heavily relies on word frequencies of the training data. For instance, if the model were trained to learn the codes “off-topic” and “on-topic” with data about electric circuits, it is extremely likely that a message talking about photosynthesis would be assigned to “off-topic,” because the word frequency distribution would not correspond to what was in the training data. Regarding language transferability, the same issue occurs. Osakwe et al. (2023) assessed their feature-based model trained on English data on Portuguese data (and vice-versa), but unfortunately, they found no favorable results. Within the same language, their results achieved reliability levels up to $k = 0.74$ with a human coder, but the same model performed poorly with kappas close to zero when testing in another language.

More recently, several studies have alternatively proposed deep learning algorithms to handle automated content analysis of CSCL processes. As Mou et al. (2016) states, neural networks can be trained in a transferable way because of their incremental learning nature. Deep learning approaches for extracting features from text usually rely on a mechanism known as ‘embeddings’ that is different from the feature-based approach (later in this paper we also refer to embeddings as ‘sentence encoders’ as this latter term is a more specific indication of embeddings that work with sub-word tokens and does not depend on language-specific pre-processing). When enough data is used to train the embeddings, words with similar meanings will be close to each other in the vector space. Furthermore, as they can be pre-trained and used posteriorly, embeddings have been regarded as a highly transferable component of neural networks (Mou et al., 2016). One related work that relied on embeddings is the study from Lämsä et al. (2021). They focused on automated content analysis of transcribed videos and audios of student interactions given the inquiry-based learning phases (Pedaste et al., 2015) that students are referring to on each unit of dialogue. These authors used embeddings pre-trained in Finnish data to train their model and achieve satisfactory results. Not only embeddings but entire pre-trained models that extract similar features from text during training have been used in related work. For instance, Lalínkar et al. (2022) reported a high reliability when employing a large pre-trained model to predict the clarity of affirmation in forum discussions. These authors did note the usefulness of such a tool in practice to trigger interventions, but they did not elaborate on how well their model could be applied to other contexts. The study from Fiacco and Rosé (2018) was likely the first to investigate transferability across topics. These authors focused on automated coding of transactivity, i.e., identifying reasoned contributions that refer to ideas previously mentioned in a discussion (Fiacco & Rosé, 2018). They exploited a pre-trained model originally trained on the entailment task, which according to them has similarities to transactivity detection task. Even though their approach was tested with multiple datasets, they were all in English. To our knowledge, an approach to achieving reliable transferability across different topics and languages at the same time is still lacking.

To tackle the aforementioned challenges, we argue that an automated content analysis algorithm for chat data that employs a pre-trained embedding strategy and simultaneously accounts for a window of context information is a promising approach to follow. In the next sections, we will explain how we designed and implemented such a system and how we assessed its reliability in terms of agreement of coding with human coding.

1.2. Overview of the study

The main goal of the present study is to provide evidence on the reliability, compared to human coders, and transferability of using ConSent for automated content analysis of student chats. Achieving this goal involves first looking for an algorithm configuration, including, for example, how many preceding contributions to use as contextual information and the sentence encoder (i.e., the embedding method), that returns the best reliability. As we will describe, ConSent handles text input and contextual information in a different way from algorithms previously proposed by related work. Thus, our evaluation also needs to uncover the benefits and potential limitations of using contextual information as we currently modeled. We aim also to explore how these effects vary over different dataset sizes. Moreover, as automating the coding scheme from Eshuis et al. (2019) will be the basis of our conversational agent tool, it is essential to investigate model performance (precision, recall, and F1-score) on each code individually to identify the ones that might be limiting overall performance and need further attention. After providing evidence on model performance in the main dataset, the current paper investigates how reliably the models can perform in a different topic and language with a second and smaller dataset. Therefore, our study is guided by the following research questions (RQs).

- RQ1: Which algorithm configuration yields the best reliability?
- RQ2: To what extent does contextual information improve model quality and how do the models improve with more training samples?
- RQ3: What are the models’ performances on each code?
- RQ4: How well do the models transfer to chats in another topic and language?

2. Method

2.1. Datasets description, coding scheme, and pre-processing

In this study, we used two datasets, coded with the same coding scheme, of student chats in different (physics) domains and different languages. The first and largest dataset was collected and coded by a previous study (Eshuis et al., 2019). These authors collected student chats during an online collaborative learning activity on the domain of electrical circuits. Triads of secondary students interacted in a Go-Lab (de Jong et al., 2021) environment where the specific tasks included creating an electrical circuit and an optimal power transmission network. The authors report inter-rater reliability of their coding scheme of $k = 0.76$ (Eshuis et al., 2019). The second dataset was collected and coded in the context of the current study. The learning activity that produced the second also took place in a Go-Lab learning environment; students interacted in dyads with a simulation examining radiation intensity over distance and time, given different particle types. On this second dataset, a second coder coded 20% of the samples, resulting in inter-rater reliability of $k = 0.68$. The first dataset was in Dutch, and the second one in Portuguese.

The coding scheme used in this paper is a subset of the coding scheme proposed by Eshuis et al. (2019). The overall goal of their coding scheme was to identify the argumentative quality of students’ communication. They proposed two coding levels (L1 and L2) and four coding groups in total (one for L1 and three for L2), each of which is composed of mutually exclusive codes. The L1 code refers to whether a message is “Informative”, “Argumentative”, “Critical statement”, “Answer for agreement”, “Other”. As mentioned, for L2, there are three coding groups. First, on responsiveness (L2R), the message can be “Response”, “Extension”, or “Other”. Second, on tone (L2T), it can be “Positive”, “Negative”, or “Other”. Third, on content (L2C), it can be, “Informative”, “Argumentative”, “Asking for information”, “Critical statement”, “Asking for agreement”, “(Dis)confirmation”, “Active motivating”, or “Other”. In this paper, we focus only on L1 and L2C, because these codes are more directly related to conditions for APT interventions (Michaels & O’Connor, 2015) that we intend to use for our conversational agent. Also, it seems that APT interventions are not much conditioned by knowing the responsiveness or tone of the message. Moreover, to make the coding scheme more general, reduce class imbalance, and possibly generate more reliable models, we also
relabeled chat messages, merging together some of the codes. For L1, “Other” was merged with “Off-task”. For L2C, “Critical statement” was merged with “Argumentative”, “Asking for agreement” with “Asking for information”, and “(Dis)confirmation” with “Informative”. Table 1 describes the codes focused on in the current paper.

In total, there were 21,341 coded chat messages in the Dutch dataset and 2,284 in the Portuguese one. The samples were filtered and preprocessed. First, to be included chats of a group should have at least 10 messages. Second, we only considered the first 500 characters of each message, as longer messages were primarily copied text from other sources, such as the main material or Wikipedia. Third, when a sample was eventually skipped by the human coder, we used the immediately previous code available as the corresponding one. After applying this filtering and pre-processing, 21,309 message samples from 187 student groups in the Dutch data and 2,209 from 32 student groups in the Portuguese data resulted. Finally, to later analyze transferability, we split each dataset into training and test sets, where the former contains 90% of the codes used in this study (adapted from Eshuis et al. (2019)).

To get a first impression of the data, Fig. 1 shows the frequencies of codes in the Dutch data, after pre-processing as they were coded by a human coder. On the left side, we show the absolute frequencies of codes. Notice that there is a clear imbalance in frequency over the different L2C codes. In our approach, this may lead to difficulty predicting the less frequent ones of these codes in a chat dialogue. On the right side, we show the co-occurrence of codes and their immediate previous (darker colors mean more frequent), illustrating how frequently a particular code is sequentially followed by another. Notice that it is for an L1 code it is rarer to observe a transition to a different one than to the same one, which may indicate that this coding group is highly context-dependent.

2.2. Neural network algorithm

The two coding models we develop, for L1 and L2C separately, are produced by the same neural network algorithm, illustrated in Fig. 2. A given message is ingested in two processing branches, namely “Sent” and “Con”. This separation can be justified as inputs of entirely different formats are used, i.e., one based on text and another on context information. In the “Sent” branch, the input text of a given message is processed using the sentence encoder. The selection of the sentence encoder used in our analysis is detailed later in this section. This encoded text representation passes through a fully connected layer with dropout to avoid overfitting. Here we compute an intermediate estimation of the output code by adding another fully connected layer with softmax activation. Then, this intermediate output is used to measure the “Sent” loss, i.e., the performance of estimating the code only by accessing the input text. By minimizing this loss, we force the “Sent” hidden layer to reliably match the target codes using only the sentence encoder’s capabilities.

In parallel to the “Sent” branch, the “Con” branch handles context information preceding the current message. The context information used in this paper is based on binary variables, namely code lags, i.e., the previous codes of the current message parsed as one-hot vectors, and auxiliary information. During training, the code lags are taken from the ground truth (i.e., those assigned by humans), but during inference, the previous codes are those estimated earlier to avoid a data leakage issue. The auxiliary information is composed of binary variables that help describe the context of the current message, such as whether the message contains a question mark and whether the message comes from the same user as the last one. Then, the context information complements the intermediate estimation made by the “Sent” branch and subsequently feeds another fully connected layer that produces the final output “ConSent” code estimation. With this structure, we aimed to separate features of heterogeneous dimension spaces, i.e., sentence embeddings and context information. This separation into branches using an intermediate code estimation might lead the model’s weights on each layer to be more coherently distributed within each other compared to if we directly concatenate the sentence encoder’s output vector to the context information. Finally, the loss to be optimized is the sum of both “Sent” and “ConSent” losses. The target and output codes are also parsed as one-hot vectors. The Adam optimizer is used in combination with categorical cross-entropy as the loss function, using the Keras (Chollet, 2015) and TensorFlow deep learning implementation framework.

2.3. Algorithm configuration

The algorithm configuration is composed of so-called hyperparameters which specify in detail the neural network algorithm. To answer RQ1, for each code separately, we applied grid-search with the Dutch training data (i.e., circuits-NL-train) to find the best configuration among the 72 option combinations possible from Table 2. During the algorithm configuration experiment, circuits-NL-train is further split into 80% for training and 20% for validation. The validation set is used to decide which is the best model, considering the highest Cohen’s $k$. Different algorithm’s learning rates and batch sizes were not included in the algorithm configuration experiment as we empirically found suitable values of $10^{-3}$ and 512, respectively. The models’ weights of the best epoch are then restored and utilized in the saved trained model.

As mentioned, we test different sentence encoders to empirically find the one that is more suitable for our data. The pre-selection of sentence encoders for this study was based on the following procedure. First, we accessed TensorFlow Hub, a platform where pre-trained embeddings and models are made available by registered publishers for different problem domains, such as text, image, video, and audio. In the search bar, we entered the keyword “Dutch”, as our main dataset is in Dutch, and found four results for text embeddings. The results included options that work with multiple languages, so we considered only those that also work with Portuguese, as our second dataset is in Portuguese. Some models published in the Hub require computationally expensive GPU infrastructure to be used, so we filtered only those that can be processed by ordinary GPU infrastructure with no difficulties. This filtering is because we prioritize the accessibility of using and training fast and lightweight ConSent models to use in our conversational agent prototype, and also considering usability for other researchers or developers without access to robust computing infrastructure. As a result, we

<table>
<thead>
<tr>
<th>Coding group</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>Domain (DOM)</td>
<td>Students are discussing domain-related topics, responding to questions, or domain-related information.</td>
</tr>
<tr>
<td></td>
<td>Coordination (COO)</td>
<td>Students are discussing how to complete the learning tasks, discussing who does what, where they are done, or how to go in the learning environment.</td>
</tr>
<tr>
<td></td>
<td>Off-task (OFF)</td>
<td>Students are discussing something unrelated to the domain or task or clearly responding to something unrelated to the domain or task.</td>
</tr>
<tr>
<td>L2C</td>
<td>Informative (IN)</td>
<td>Informing other students of knowledge/facts, actions, or personal states.</td>
</tr>
<tr>
<td></td>
<td>Argumentative (AR)</td>
<td>Argumentative utterance aiming at clarifying, reasoning, interpreting, stating conditions, or drawing conclusions.</td>
</tr>
<tr>
<td></td>
<td>Asking for information (AI)</td>
<td>Asking for explanation, confirmation, clarification, or assistance.</td>
</tr>
<tr>
<td></td>
<td>Active motivating (AM)</td>
<td>Encouraging (a) group member(s) to participate or to take action.</td>
</tr>
<tr>
<td></td>
<td>Other (NOS)</td>
<td>None of the specified above can be interpreted from the message.</td>
</tr>
</tbody>
</table>
selected two pre-trained encoders to investigate which one may lead to better models in RQ1. The retrieved ones are Wiki40b-lm-multilingual\(^1\) (Guo et al., 2020), and USE-multilingual\(^2\) (Yang et al., 2019). The first one was trained with the Wiki40B dataset (preprocessed Wikipedia articles in 41 languages) in a transformer-XL algorithm architecture. The second one was trained using a convolutional neural network algorithm architecture on 16 languages across multiple datasets and tasks, such as question and answering pairs mined from widely used web discussion forums and text entailment with the SNLI corpus. The encoders’ internal parameters remained frozen during training; we use them statically as is.

### 2.4. Evaluation

Once we find the algorithm configuration that retrieved the highest Cohen’s \(k\) in the analysis of RQ1, we proceed using it to analyze RQ2, RQ3, and RQ4. To have a more representative reliability estimation, we performed 5-fold cross-validation (CV) and averaged the results. When analyzing RQ2, Cohen’s \(k\) was measured for overall reliability. We compared the average performance of “Sent”-based versus “ConSent” estimations. Also for RQ2, we applied the same procedure given

---

\(^1\) https://tfhub.dev/google/wiki40b-lm-multilingual-64k/1.

different sample sizes of training data (20%, 40%, 60%, 80%, and 100%). For RQ3, we measured reliability of individual codes using the metrics precision, recall, and F1-score. These metrics are employed to evaluate individual performance per class whereas Cohen’s $k$ is used for overall performance. High recall for a specific code means the model makes fewer false negative mistakes (i.e., those incorrectly coded as not part of a code) and high precision means fewer false positive mistakes (i.e., those incorrectly coded as part of the code). The F1-score is the harmonic mean of precision and recall and is usually helpful when both error types are equally important in the analysis.

Then, to answer RQ4, we evaluated the transferability of our coding models using three approaches. First, models are trained in one domain and language and evaluated in both test sets (i.e., radiation-PT-train) to improve reliability in the corresponding test set (i.e., radiation-PT-test). For that, two other approaches were considered. One by training from scratch using both circuits-NL-train and radiation-PT-train datasets together. Another by fine-tuning the models trained only with circuits-NL-train using the radiation-PT-train data. These two approaches were designed to help evaluate transferability, or whether models for one domain and language benefit from training in another. To refer to the models, from now on, we adopt the following aliases.

- ConSent-L1-NL and ConSent-L2C-NL for the main models trained with circuits-NL-train;
- ConSent-L1-NL + PT and ConSent-L2C-NL + PT for the models trained with both training datasets;
- ConSent-L1-NL*PT and ConSent-L2C-NL*PT for the models fine-tuned with radiation-PT-train.

More specifically, the fine-tuning approach was made as follows. We created a new model with the same configuration as found in RQ1 (except for the algorithm’s learning rate), initialized the two hidden layers’ weights using the ones from the pre-trained model, and started training. One of the main challenges here was to avoid the effect known as catastrophic forgetting (McCloskey & Cohen, 1989) associated with the algorithm’s learning rate (Sun et al., 2019), in which a fine-tuned model erases knowledge acquired in pre-training. To alleviate this problem, we performed a preliminary experiment and empirically found that a learning rate of $10^{-5}$ yielded a significantly smaller drop in performance when compared to the $10^{-3}$ used originally during pre-training. Due to the much lower learning rate, the models were fine-tuned for around 1,000 epochs to slowly achieve a flat learning curve without forgetting too much of the previous training.

3. Results

3.1. RQ1: Which algorithm configuration yields the best results?

The algorithm configuration found to lead to better models is shown in Table 3. Regarding the time performance of the best models, we measured that inference takes in average 0.36s (L1) and 0.33s (L2C) using a simple 4 GB RAM dual-core CPU machine. We included the best configuration for each sentence encoder. Note that, for both code groups, USE-multilingual generated better performance by a significant amount. The reason is likely to be related to the fact that USE-multilingual was trained on question-answers pairs from web forums, whereas the Wiki40b-lm-multilingual was mainly based on Wikipedia texts, which are more structured and formal. Besides, we also found different configurations of the “Sent” hidden layer sizes for each code group. Considering only the best models, L1 was better with a smaller hidden layer, which consequently required a lower dropout rate as it is less likely to overfit, whereas L2C worked better with more neurons and a higher dropout. In both cases, the “ConSent” hidden layer size and the number of code lags were the same. Interestingly, a larger number of code lags benefitted the models of both code groups. This gives a preliminary indication that context features are beneficial, but we will examine that in-depth by analyzing RQ2.

3.2. RQ2: To what extent does contextual information improve model quality and how do the models improve with more samples?

The CV performance comparison of the intermediary (i.e., “Sent”-based) and output codes (i.e., “ConSent”-based) is shown in Fig. 3. The reliability numbers were a bit different than those shown in Table 3 because this time we analyzed performance across 5 folds of test data and averaged the results. As expected, L1 and L2C models consistently improved when using contextual information compared to only using the sentence encoder. L1 models had an average kappa of 0.60 with and 0.51 without contextual information, while L2C models had 0.63 with and 0.56 without contextual information. This upgrade from the intermediary to output codes seems more pronounced for L1 codes probably because of the high sequential correlation we pointed out when discussing Fig. 1. Also, observe that the coding models have a higher kappa for L2C than for L1. Since the “Sent”-based performance is already higher in L2C than L1, the reason might be because L2C codes can be more easily coded from the text alone without context information.

![Fig. 3. Cross-validated scores when using the sentence-based only intermediary codes (“Sent”) and the final output codes using context features (“ConSent”).](image-url)

Table 3

<table>
<thead>
<tr>
<th>Code group</th>
<th>sentence encoder</th>
<th>sent_hl_units</th>
<th>sent_dropout</th>
<th>consent_hl_units</th>
<th>lags</th>
<th>Cohen’s $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>Wiki40b-lm-multilingual</td>
<td>1024</td>
<td>0.2</td>
<td>128</td>
<td>7</td>
<td>0.2417</td>
</tr>
<tr>
<td></td>
<td>USE-multilingual</td>
<td>256</td>
<td>0.5</td>
<td>32</td>
<td>7</td>
<td>0.6308</td>
</tr>
<tr>
<td>L2C</td>
<td>Wiki40b-lm-multilingual</td>
<td>256</td>
<td>0.2</td>
<td>128</td>
<td>7</td>
<td>0.3246</td>
</tr>
<tr>
<td></td>
<td>USE-multilingual</td>
<td>1024</td>
<td>0.8</td>
<td>32</td>
<td>7</td>
<td>0.6493</td>
</tr>
</tbody>
</table>
When using more data from the same domain to train these models, we see that overall they do not improve much more after 60% of the total samples used (See Fig. 4), which corresponds to around 11,000 samples. Even with 40% (around 8,000 samples), the models are already quite close to their final performance. It suggests that simply adding more data from the same domain and language would not be a solution. Implicitly, it also suggests that training with the Portuguese data, that has around 2,200 samples, would not be enough to build a reliable model from scratch, as with 20% of the Dutch data (around 4,000 samples), the models already perform below acceptable reliability levels. Also, another result we can notice from Fig. 4 is that “Sent”-based performance (blue curve) in L1 peaks with 80% of samples. But, in L2C, it keeps steadily growing, again confirming that there are more opportunities for the sentence encoder branch to predict L2C codes than L1 ones, at least in this dataset.

### 3.3. RQ3: What are the models’ performances on each code?

We measure the precision and recall to pinpoint the models’ CV performance on each code. The results are described in Table 4. Note that, on the one hand, L1 codes have reasonably balanced scores, which may be related to the balanced amount of samples in each (see Fig. 1). For DOM, there is an equilibrium between precision and recall, but the model seems to have a tendency to make more false positive mistakes towards COO (higher recall) and more false negative mistakes towards OFF (higher precision). On the other hand, the scores on L2C codes were not that balanced. The L2C model performed poorly on the AR and AM codes, those with the lowest amounts of samples, with F1-score of 0.28 and 0.38, respectively. AR and AM model’s precision is significantly higher than recall, meaning that the model prefers to make a false negative kind of mistake than a false positive one. In other words, the model avoids risking an estimate on these two codes, as there is not so much data describing them. As for IN (i.e., the L2C majority class), the opposite happens; recall is much higher than precision, which means the model has a tendency to take risks and have a higher rate of false positive mistakes. As we notice in RQ2, the L2C model still has potential to improve, now knowing that this might be achieved by having more AR and AM samples.

### 3.4. RQ4: How well do the models transfer to chats in another domain and language?

As explained previously, we analyze RQ4 with three approaches. The results are detailed in Table 5. In the first approach, we checked the difference in performance of a model trained in one domain and language in another different domain and language. The results are the two first rows of each code group block in Table 5. The results show that the ConSent-L1-NL model dropped from 0.71 in the original domain and language to 0.52, and the ConSent-L1-PT from 0.68 to 0.49. Similarly, the ConSent-L2C-NL dropped from 0.70 to 0.44, and, unexpectedly, ConSent-L2C-PT went up from 0.53 to 0.54. The reason might be that, even though the Portuguese training data was not enough to learn the codes reliably, it had a stable understanding of the codes to the point of performing similarly well on the other dataset. Excluding this last case, models drop considerably when they are evaluated in a different domain and language but note that they still keep close to the 0.5 level.

Then, we assessed the two other approaches. On the second approach, we can see that by training with both datasets together (denoted by the NL+PT suffix), the L1 model does not perform as well as before, neither in the Dutch dataset nor in the Portuguese one. On the other hand, the L2C model performs quite well when trained on both datasets at the same time. This difference is likely because the L1’s DOM code is more confusing in this mode of training, as the words employed in the chat messages are different in each dataset and the model struggles as a result. On the third approach, by fine-tuning (denoted by the NL*PT suffix), the model achieved the highest reliability in the Portuguese test set with a kappa of 0.76 for the L1 codes and 0.69 for the L2C codes. But it drops more heavily on the Dutch test set that corresponds to the original domain and language on which the model was trained.

### 4. Discussion

The major challenge in automated content analysis is to create a tool

![Fig. 4](image-url) Cross-validated median performance of models trained with different sample sizes, from 20 to 100% of the training set. In this dataset, for both models the improvement from using 20% and 100% is considerable, but not much after 60%, which corresponds to around 11,000 chat messages.

---

Table 4

<table>
<thead>
<tr>
<th>Code group</th>
<th>Code</th>
<th>Precision M (SD)</th>
<th>Recall M (SD)</th>
<th>F1-score M (SD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>DOM</td>
<td>0.75 (0.04)</td>
<td>0.74 (0.06)</td>
<td>0.75 (0.02)</td>
</tr>
<tr>
<td></td>
<td>COO</td>
<td>0.68 (0.02)</td>
<td>0.74 (0.03)</td>
<td>0.71 (0.01)</td>
</tr>
<tr>
<td></td>
<td>OFF</td>
<td>0.78 (0.05)</td>
<td>0.71 (0.09)</td>
<td>0.74 (0.06)</td>
</tr>
<tr>
<td>L2C</td>
<td>IN</td>
<td>0.74 (0.02)</td>
<td>0.84 (0.01)</td>
<td>0.79 (0.01)</td>
</tr>
<tr>
<td></td>
<td>AR</td>
<td>0.38 (0.03)</td>
<td>0.22 (0.03)</td>
<td>0.28 (0.02)</td>
</tr>
<tr>
<td></td>
<td>AI</td>
<td>0.76 (0.04)</td>
<td>0.74 (0.03)</td>
<td>0.75 (0.02)</td>
</tr>
<tr>
<td></td>
<td>AM</td>
<td>0.47 (0.04)</td>
<td>0.32 (0.03)</td>
<td>0.38 (0.02)</td>
</tr>
<tr>
<td></td>
<td>NOS</td>
<td>0.79 (0.05)</td>
<td>0.72 (0.06)</td>
<td>1.75 (0.05)</td>
</tr>
</tbody>
</table>

Table 5

<table>
<thead>
<tr>
<th>Model</th>
<th>Test dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConSent-L1-NL</td>
<td>circuits-NL-test</td>
</tr>
<tr>
<td>ConSent-L1-PT</td>
<td>circuits-NL-test</td>
</tr>
<tr>
<td>ConSent-L1-NL + PT</td>
<td>circuits-NL-test</td>
</tr>
<tr>
<td>ConSent-L1-NL*PT</td>
<td>circuits-NL-test</td>
</tr>
<tr>
<td>ConSent-L2C-NL</td>
<td>radiation-PT-test</td>
</tr>
<tr>
<td>ConSent-L2C-PT</td>
<td>radiation-PT-test</td>
</tr>
<tr>
<td>ConSent-L2C-NL + PT</td>
<td>radiation-PT-test</td>
</tr>
<tr>
<td>ConSent-L2C-NL*PT</td>
<td>radiation-PT-test</td>
</tr>
</tbody>
</table>
that human coder creates a valid coding, potential threats to validity can be addressed by seeking to increase this reliability (Rosen et al., 2008). While CSCL researchers have suggested that reliable models for automated content analysis can serve to feed real-time dashboards and trigger instructional interventions, not many studies discussed the practical challenges for that. Regarding the use of such tools in the development of conversational agents, we highlighted three additional challenges that needed further research. First, as student chats are highly context-dependent, the algorithms should account for context information in order to produce reliable models. Second, as it is not feasible to train a separate model for each domain, the models should be transferable to other (similar) domains. Third, as a requirement of our conversational agent to increase usability on different languages, the model should be transferable to other languages as well. We proposed and evaluated ConSent as an algorithm to handle these challenges.

An investigation of related work reveals that comparisons of reliability levels across studies are complicated because multiple factors interfere. Among other factors, different private datasets are used, and the dataset size and quality of written text (e.g., abbreviations and lack of punctuation) varies considerably and can influence models’ performance. Even though Cohen’s k is predominantly employed to measure overall reliability, the evaluation approach and the coding scheme’s complexity interfere when comparing the reliability levels from different studies. Nonetheless, there are ranges of reliability acknowledged as acceptable. A Cohen’s k of at least 0.7 is recommended for highly reliable content analysis, even though a kappa of 0.6 is already considered “substantial” agreement between a machine-learned classifier and a gold standard (McLaren et al., 2010). In the current work, we did not aim to compete with other studies on reliability, but rather to assess the capabilities of ConSent and discuss its applicability to help conversational agents to trigger chat interventions in a group conversation.

The results reported in this paper demonstrate that ConSent models can achieve a substantial level of reliability for automatically coding student chats. By analyzing RQ1, we found the best algorithm configuration among the 108 possibilities tested and then employed it to answer the other RQs. Regarding the sentence encoder chosen, we found that using USE-multilingual yields a considerably higher reliability compared to Wiki400-blm-multilingual. The reason is likely because the former was pre-trained on questions-answers pairs from web forums, which have more informal communication, and the latter on Wikipedia articles. Also, we observed that the best models relied on more contextual information, since employing 7 code lags yielded the best results. While examining RQ2, we confirmed that contextual information as currently modeled does improve reliability by a significant amount. With cross-validation, we found that models performed in average with a reliability of $k = 0.60$ for L1 codes and $k = 0.63$ for L2C codes. On the one hand, it seemed that the L1 model gained more from contextual information than the L2C model. On the other hand, the L2C model had higher reliability than the L1 model because the sentence encoder was more effective at detecting the L2C codes given the text only. Thus, we argue that one complemented the other. Also, inspecting RQ2, we found that adding even more data points from the same domain would not directly benefit the models in a significant way.

Last but not least, the analysis of RQ4 revealed encouraging results. To assess transferability to another domain and language, models were trained on student chats about electrical circuits in Dutch and tested on chats about radiation in Portuguese, and vice-versa. We observed that the models dropped almost 30% in performance, but still performed moderately with a Cohen’s k of approximately 0.5. These results reinforce the idea that the models are transferable because the knowledge acquired in one domain and language explicitly helped in another. Subsequently, we examined how the models improved (1) when we use the two datasets during training, and (2) when we fine-tuned a model trained with the Dutch data using the Portuguese data. First, we found that for the L2C model, it seems that training with both datasets together resulted in a stable performance on the test sets, but that did not happen for the L1 model. The reason is likely because, in L1, the code DOM (i.e., “Domain”) has two different sets of keywords to be associated with, i.e., about circuits and about radiation. Second, fine-tuning resulted in improvements for both L1 and L2C. The highest reliability of the Portuguese dataset in both L1 and L2C models was found with the fine-tuning approach. Therefore, in order to achieve a substantially reliable transfer of ConSent models, it might be required to fine-tune with a few more samples from the target domain and language.

4.1. Implications

As mentioned previously, automated content analysis yields opportunities from both scientific and practical perspectives. After analyzing the results, we argue that ConSent models can be reliably used by researchers to automatically code student chats in domains similar to the ones utilized in this paper. A simple double-checking on the codes generated by ConSent to correct the mistakes is still necessary as high reliability is crucial for experimental studies. Our results also show that multilingual sentence encoders and contextual information, as currently modeled, can serve as transferable features for automated content analysis algorithms. Consequently, conversational agent designers can build on our findings to further explore algorithm configurations that achieve even higher reliability on data from various topics and various languages when creating AI-based learning systems with a wide usability range.

From a practical perspective, we see a number of applications for ConSent to help improve education. First, in line with the original goal of our work, ConSent can be the basis for conversational agents that are intended to improve discussions between students in collaborative learning arrangements. In this use case, ConSent can be used to analyze the ongoing chat of students in real-time and, based on this analysis as well as a set of rules using the identified categories, interventions by conversational agents can be triggered. If a firm theoretical basis is used for these interventions, like in our case APT (Michaels & O’Connor, 2015), student acceptance of these interventions and consequently an improvement of the dialogue quality could be expected. We also expect that by using a solid, theory-based, chat analysis that can guide interventions, a conversational agent will intervene more naturally, making students receptive for its feedback. Of course, this all needs to be put to the test by subsequent work. In this regard, our approach has the advantage that ConSent models can be used to build larger sets of prototype conversational agents, as they can be directly transferred to other (similar) domains and languages with a moderate reliability level (or fine-tuned with a small dataset to further increase reliability). A second use case concerns interventions in student dialogues which are not presented to students in real time, but rather to teachers, retrospectively. Specifically, ConSent can be used to create a dashboard presenting a graphical overview of students’ conversations so that teachers may identify overall patterns (e.g. strengths and shortcomings) and discuss these in class. Previous studies also suggest creating teacher dashboards for displaying real-time information (Chen et al., 2015) and ConSent could be used similarly. A third use-case is when ConSent is used to create a dashboard for students to reflect on their dialogues. To design
Such dashboards, subsequent work has to manipulate the codes’ frequency produced from ConSent to decide on measures for the quality of students’ dialogues based on a pedagogical conceptual model (e.g., Tegos et al., 2021). For a more comprehensive list of applications and a deeper discussion on students’ receptivity for AI-based learning systems, we refer to Chiu et al. (2023).

4.2. Limitations

Although the results presented are promising, this work has some limitations. First, we designed models based on computationally inexpensive sentence encoders so that researchers could train them in standard computers. For even better reliability, it would be wise to test larger pre-trained language models. Also, USE-multilingual chosen only handles 16 languages (Arabic, Chinese-simplified, Chinese-traditional, English, French, German, Italian, Japanese, Korean, Dutch, Polish, Portuguese, Spanish, Thai, Turkish, and Russian). More languages would be necessary for even broader usability. Furthermore, in related work, other kinds of contextual information were proposed that would potentially increase performance of our models. For instance, Kovanovic et al. (2016) utilized a LSA metric to measure semantic similarity between the text sample and a corresponding Wikipedia page content about the domain. Another potential limitation was that we did not leverage data curation to correct or improve how our data was coded. Moreover, we did not include a comparison with a baseline approach, although we tested different sentence encoders to pick the best one. Thus, further upgrades in reliability might be achieved with two approaches or a combination of both. The first is model-centric, for example, by including other contextual features that can be relevant and by using a more powerful sentence encoder, even fine-tuning its internal weights. The second is data-centric, for instance, by augmenting the dataset with translations or by curating it for samples that are mistakenly coded. Conversational agent designers must also account that the models did not perform well in some of the codes and a post-processing step might be required in order to use those in triggers. Moreover, it is important to highlight that since we only tested in two datasets of similar domains within science education, our findings might not generalize to outward domains that does not contain science-related keywords in the chat messages.

5. Conclusion

Automated content analysis of student chats is an interesting research topic that can potentially support both research and practice. A new machine learning-based system for automated coding of student chats called ConSent was presented in this paper. The core of this system is a new algorithm that uses contextual information and sentence encoding to provide reliable estimates of the code of a chat message. Results indicated that models are able to perform with substantial reliability levels and transfer reliable coding of chats in a similar domain and different language. Therefore, the approach may be suitable for the development of a conversational agent. The particular coding scheme used to train our models should be regarded as an example of what can be achieved with such a tool, and we encourage researchers to apply the same procedure described in this paper to automate a different set of codes. However, to make chat coding more reliable, more advanced learning models may be needed. As a recommendation for future work, researchers can address both model-centric and data-centric approaches, as discussed previously. Finally, it should be noted that although we have tested our approach on two domains and two languages, much can be learned by applying ConSent to student chats in other educational contexts and languages. Also, by showcasing what such models can do, we expect that educational practitioners can effectively trust and rely on them to adapt instruction in a data-driven manner.

Declaration of competing interest

The authors declare that they have no conflict of interest.

References


