Detecting Exclusive Language During Pair Programming

Solomon Ubani, Rodney Nielsen, Helen Li
Department of Computer Science and Engineering, University of North Texas, USA
solomonubani@my.unt.edu, rodney.nielsen@unt.edu, helenli@my.unt.edu

Abstract

Inclusive team participation is one of the most important factors that aids effective collaboration and pair programming. In this paper, we investigated the ability of linguistic features and a transformer-based language model to detect exclusive and inclusive language. The task of detecting exclusive language was approached as a text classification problem. We created a research community resource consisting of a dataset of 40,490 labeled utterances obtained from three programming assignments involving 34 students pair programming in a remote environment. This research involves the first successful automated detection of exclusive language during pair programming. Additionally, this is the first work to perform a computational linguistic analysis on the verbal interaction common in the context of inclusive and exclusive language during pair programming.

Introduction

Remote pair programming (RPP) is a practice in software engineering where two programmers that are in different physical locations collaborate to develop software virtually. RPP shares similarities with traditional co-located pair programming where the person typing the code (the driver) and the person watching that coding to spot any errors or ways to improve the code (the navigator) switch roles regularly. Optimally, both pair partners verbalize their thoughts, questions, and ideas (Hughes et al. 2020). Pair programming, when done effectively, is known to improve code quality, communication, and coding skills of both pair programmers. Pair programming, unfortunately, is not always done effectively. The most significant obstacles to effective pair programming include poor communication and imbalanced power or role dynamics (Begel and Nagappan 2008). This work aims to provide an automated and intelligent support model that in the future can be integrated into systems to improve communication and balance role dynamics during collaboration.

Villela and Cordova (2015) noted the importance of inclusive participation during pair programming and stated that an experienced navigator is good at communicating their intentions and actions using inclusive language (for example, saying "us" and "we", rather than "I" or "you") as much as possible. This invites the navigator to introspect on what motivated the driver's intentions and actions that they (the navigator) might disagree with. Frykedal and Chiriac (2018) noted that students' inclusive participation requires that the students invite each other into the group discussions. Pronoun usage that implies inclusivity suggests that every partner is actively involved in the collaboration and can change the working dynamic amongst collaborators. However, both Penycook (1994) and Pantelides & Bartesaghi (2012) discussed how some pronouns can both imply inclusivity and exclusivity depending on the context. For example, in the sentence "we computer science majors code better", the pronoun "we" could be exclusive or inclusive depending on whether their partner is a computer science major or not. This shows that typically inclusive pronouns such as "we" or "us" could imply exclusivity in some contexts. Similarly, pronouns like "I" and "you" generally imply exclusivity (I will let you do that alone), but there are instances where "I" and "you" could imply inclusivity ("you were right").

The main contributions of this research are:

- This is the first work to successfully automate the detection of exclusive language in pair programming dialogue.
- This is the first work to perform a computational linguistic analysis on inclusive and exclusive language.
- The high-quality publicly-accessible dataset developed as part of this research is the first dataset of pair programming dialogue labeled to indicate utterances with exclusive, inclusive, and neutral language.

Related Work

Michailidis et al. (2018) developed an Interaction Analysis (IA) toolkit for blogs that does real-time analysis of the social interactions that occur during collaboration to

generate graph visualizations. The graphs generated by the IA toolkit aid teachers and students to track and regulate the collaboration process. Results show that the IA graphs generated by the system improve participation and self-regulatory competencies of collaborating students. Tegos et al. (2014) developed a system called MentorChat, which uses a conversational agent that is domain-independent to help trigger more productive dialogue from collaborating students. The agent monitors group dialogues to recognize avenues for intervention. In each group assignment, the agent provides domain knowledge as interventions that either target the "weak" student or the whole group. The student who introduced a concept into the discussion is a strong student whereas the other student(s) are referred to as weak students. The results show that an agent that targets "weak" students outperforms an agent that targets the whole group in increasing productive dialogue.

Casamayor et al. (2009) developed an intelligent collaborative agent that succeeds in identifying detrimental peer interaction. First, the agent models users with data gathered from their interactions with the system to obtain a user model. Next, the user model is combined with the student's learning style (defined by a pretest) with the aim of intelligently detecting conflicts using predefined rules. Finally, the agent adaptively notifies the teacher when conflicting situations that require intervention are detected. The teacher chooses whether or not to intervene. Results show that the system was very effective in detecting conflicts and teacher intervention to resolve conflicts was needed most of the time.

Ubani and Nielsen investigated the feasibility of using a transformer-based language model to detect micromanagement (Ubani and Nielsen 2021) and exploratory talk (Ubani and Nielsen 2022) during student pair programming sessions. Their results show that detecting micromanagement and exploratory talk is feasible using pretrained language models. They noted that the models could be integrated into a collaborative intelligent tutoring system to improve students' learning during collaboration.

In this work, we detect the use of exclusive language in pair-programming dialogues within the context of student assignments. Jeong et al. (2016) noted that being aware of the consequences of disruptive collaborative situations (in this case, the use of exclusive language) and addressing them can help improve collaborative dialogue and overall collaborative performance. Ubani and Nielsen (2022) emphasized the need for more systems that analyze dialogue and provide verbal interaction support during collaboration. If Machine Learning (ML) is successfully applied for detecting exclusive language, one possible use case of the resulting model is that it can be integrated into Collaborative Intelligent Tutoring System (CITS). The resultant CITS will facilitate pair programming to help achieve inclusive participation by detecting exclusive language and, where appropriate, nudging pair programmers in real time to use inclusive language while collaborating.

Dataset

We approach the task of detecting exclusive language as a supervised ML problem. Therefore, we need a dataset of relevant dialogue annotated as inclusive, exclusive, or neutral language, but no such corpus exists. Consequently, we created a dataset by labeling the transcripts of RPP assignments in a computer science graduate course.

Data Collection and Participants

The data was collected from 34 participants who were all graduate students at a large university in the United States and had prior programming experience. The students were given programming assignments in a Machine Learning class. The programming assignments were coded using the Python programming language. Specifically, the assignments were on the Perceptron, Logistic Regression and Artificial Neural Networks.

The 34 collaborating students were assigned different partners that were chosen at random for each of three RPP-based assignments. This resulted in a total of 17 pairs x 3 assignments = 51 distinct pair collaboration sessions. In each pair programming homework, the two students were in different physical locations and collaborated remotely using ZOOM to video chat and Google Colab to share their code.

Analysis and Labelling

Label	Definitions
`Inclusive Language`	Utterance shows evidence that both programmers are actively involved in the collaboration
`Exclusive Language`	Utterance shows evidence that one partner is not actively involved/invited into some part of the collaboration
`Neutral`	Utterance is neither Inclusive nor Exclusive

Table 1: Labels used for detecting exclusive language

Label	Hw 1	Hw 2	Hw 3	Total
`Inclusive Language`	1217	1427	2709	5353
`Exclusive Language`	198	109	379	686
`Neutral`	7418	9741	17292	34451
Total utterances	8833	11277	20380	40490

Table 2: Number of utterances by label and homework

The transcripts for the corpora were obtained by automatically transcribing the collaborative sessions using the ZOOM automatic transcription feature. Five collaborative sessions that failed to auto-transcribe were excluded from the corpora. Next, the transcripts were split into utterances. For this work, an utterance is defined as (1) a complete sentence (including interjections, exclamations and back-channel) or (2) an incomplete sentence followed by a pause that lasts a minimum of 500 milliseconds. A total of 40,490 utterances were manually labeled according to robust annotation guidelines (elaborating the definitions in Table 1). Our annotation guidelines and definitions of exclusive and inclusive language were built from research on inclusive pronouns from Penycook (1994) and Pantelides & Bartesaghi (2012) and research on inclusive questions from Oyeleye & Ayodele (2012). Pantelides & Bartesaghi (2012) performed discourse analysis on transcripts of collaborative conversations to highlight how a pronoun can be inclusive, exclusive or neutral depending on the context they are used in. Similarly, Oyeleye & Ayodele (2012) performed discourse analysis on legislative sessions and highlighted the types of interrogatives that are inclusive, exclusive and neutral. Our annotation guideline contained multiple examples of utterances from the aforementioned research studies that were inclusive, exclusive and neutral. For example, multiple examples of the pronoun 'you' were discussed in the inclusive, exclusive and neutral contexts. Similarly, multiple examples of elicitation and directive interrogatives were discussed in the inclusive, exclusive and neutral contexts. The annotators were trained on the guidelines and were instructed to label utterances following the guidelines. A breakdown of the number of utterances labeled as `Inclusive Language`, `Exclusive Language` and `Neutral` for each homework assignment is shown in Table 2.

Two annotators independently followed the annotation guidelines and labeled each utterance in the transcripts. The annotators achieved an overall Cohen's kappa (Landis & Koch, 1977) of 0.88 and inter-annotator F_1-score of 0.97. To calculate the inter-annotator F_1-score, annotator-1's recall relative to annotator-2 is annotator-2's precision relative to annotator-1 and vice-versa. At a more fine-grained level, the annotators achieved a Cohen's Kappa inter-rater reliability score of K=0.95 for `Inclusive Language` and K=0.57 for `Exclusive Language`, interpreted as almost perfect and moderate agreement, respectively, by Landis and Koch (1977). The inter-annotator F_1-score for detecting `Inclusive Language` was 0.96 and for detecting `Exclusive Language` was 0.58.

Annotation disagreements (between the two raters) were adjudicated by a third, more proficient, analyst. Table 3 presents the performance of the annotators against the adjudicated gold standard data. With the assumption of independence in errors made by the annotators, the extremely high recall achieved by each annotator relative to the gold standard suggests that only 0.2% (0.04*0.05) of utterances with *exclusive language* were missed by both annotators and, hence, not identified in the final gold-standard dataset. Therefore, given independence of annotator errors, this implies that the final adjudicated gold-standard dataset should be of high quality, identifying around 99.8% of utterances containing *exclusive language*.

Annotator	Metric	`Exclusive Language`	`Inclusive Language`
Annotator 1	Precision	0.71	0.99
	Recall	0.96	0.99
	F_1-score	0.82	0.99
Annotator 2	Precision	0.53	0.96
	Recall	0.95	0.98
	F_1-score	0.68	0.97

Table 3: Performance of the annotators relative to the adjudicated gold standard data

In the following sections, we develop context-sensitive machine learning models to successfully automate the detection of exclusive language in pair programming dialogue. We first discuss how we partition our data into training (for model development), tuning (to choose hyperparameters) and testing (to evaluate our model) datasets. Then, we discuss our baseline (see Baseline) and transformer-based machine learning model (see Model Training and Hyperparameters). Next, we discuss the error analysis of our model to investigate avenues for improvements in our revised model (see Error Analysis). Finally, we discuss our revised model and a novel approach for evaluating the final model.

Dataset Partitioning: Training, Tuning, and Testing Datasets

We partitioned the data by student, splitting the 34 students randomly. The training dataset contains the utterances of 20 students, the tuning dataset contains the utterances of 7 students, and the testing dataset contains the utterances of the remaining 7 students. Additionally, our final results are based on training and tuning our model on the utterances from two homework assignments and testing on utterances from the third homework, iterating over the three assignments in what is essentially a leave-one-(homework)-out evaluation. Thus, the results on the testing dataset(s) reported in this paper reflect the expected results on production data–new dialogue from new students and

new homework assignments, but in a similar remote pair-programming context. The exact number of utterances in each dataset are shown in Table 4. Our final training is performed on the combined training and tuning datasets.

	Training	Tuning	Testing	
HW	Utterances	Utterances	HW	Utterances
2,3	18825	6352	1	1692
1,3	18396	5357	2	2712
1,2	11761	3945	3	3768

Table 4: Number of utterances in each dataset

Baseline

We built a term-frequency inverse document frequency (TF-IDF) *baseline* model. TF-IDF is a method used to determine the importance of words in a *text* based on their frequency in that *text* and in the overall corpus (Jones 1972). For this baseline only, the dataset was preprocessed by first converting the dialogues to lowercase and removing punctuation and stopwords. After preprocessing, features were extracted as TF-IDF vectors and used to train a support vector machine (SVM) classifier. We used grid search to tune the SVM's hyperparameters via cross-validation on the tuning data. The baseline model achieved a macro-average F_1- score over the three *testing* datasets of 0.23 for detecting `Inclusive Language` and 0.09 for detecting `Exclusive Language`.

Methodology

The task of detecting *exclusive language* was approached as a text classification problem by fine-tuning pretrained language models–a form of transfer learning. Pretrained language models have been widely considered one of Natural language Processing's (NLP's) biggest successes of transfer learning (Raffel 2019). The models are pretrained on massive corpora for language modeling tasks and can then be fine-tuned end-to-end for new downstream tasks. We fine-tuned the pretrained BERT model to classify remote pair programming (RPP) utterances according to their labels using a combination of the training and tuning datasets and reported results based on the testing datasets.

We determined the hyperparameters for fine-tuning BERT based on performance on the tuning dataset. Our resulting hyperparameters were a training time of three epochs, batch size of 32, the AdamW optimization function with a learning rate of 5×10−5, the categorical cross entropy loss function, and freezing the first six of BERT's 12 blocks before fine-tuning.

Results from Fine-tuning BERT

The resulting model obtained from fine-tuning BERT had a weighted average F_1-score of 0.91. The model had macro-average F_1-scores (over the three homework assignment testing datasets) of 0.97 for detecting `Inclusive Language` and 0.45 for detecting `Exclusive Language` (see Table 5 for more details).

Error Analysis

We conducted an error analysis on all 464 instances misclassified on the tuning datasets. The analysis consisted of two parts. The first analysis consisted of a manual review and categorization of the errors, while the second utilized pointwise mutual information (PMI) to investigate potential linguistic constructs correlated with the errors.

Part 1: Manual Review Error Categories

In the first analysis, we analyzed each misclassified instance, noting their peculiarities and the possible causes of misclassification. We aggregated our findings into five error categories which consisted of errors that stemmed from speech disfluencies, annotation errors and automatic speech recognition errors.

	Inclusive language			Exclusive language		
Experiment	Precision	Recall	F_1-Score	Precision	Recall	F_1-Score
Baseline (SVM + TF-IDF)	0.402	0.161	0.230	0.172	0.064	0.093
Human Inter-Annotator	0.971*	0.952*	0.961	0.681*	0.512*	0.584
Human Annotator vs. Gold Standard (averaged)	0.983	0.990	0.986	0.622	0.961	0.755
BERT	0.971	**0.963**	**0.967**	0.541	0.422	0.474
BERT + Linguistic Features	0.982	0.952	**0.967**	0.613	**0.431**	**0.505**
BERT + Linguistic Features (Confidence score > 90%)	**1.000**	NA	NA	**0.700**	NA	NA

Table 5: Results on the dataset reporting macro-average (over three homework assignments) precision, recall and F_1-score

Repeated Words or Phrases in an Utterance (Disfluencies in Spontaneous Speech)

Example**:** Stick stick it in there without the Irish class

FIX: Can be corrected with a regular expression that can detect duplicate words or phrases[1]

Correcting for repeated word disfluencies in the utterance enabled the model to correctly predict that the above utterance, for example, was `Exclusive Language` rather than the previous erroneous prediction of `Neutral`. We observed about 5% of utterances during our error analysis had repeated words or phrases.

Incorrect Word Case in an Utterance Transcription

Example**:** The Click did not work

FIX: Can be corrected with a true casing tool, for example: https://pypi.org/project/truecase/

Correcting the case of the utterance enabled the model to correctly predict that the above utterance was `Neutral` rather than the previous erroneous prediction of `Exclusive Language`. We observed about 30% of utterances during our error analysis had at least one word with an incorrect case.

Errors in Human Annotation

During error analysis, we discovered that some utterances were labeled erroneously. The model correctly classified the utterances, but the labels in the ground truth were incorrect.

Example: I just saved it

Correcting the annotation of the utterance from `Exclusive Language` to `Neutral` facilitated accurate evaluation of the model on the utterance. Incorrect labels in the training sets could also be the cause of some misclassifications on the test set. We observed about 8% of utterances during our error analysis were labeled incorrectly.

Automatic Speech Recognition (ASR) Errors

In addition to mistakes in case, some errors in transcription in the dataset could have led to incorrect classifications made by the model since the utterances sometimes become unintelligible.

Example: Now all we need is you need to

Correcting the transcription of the utterance to "Now all we need is U2" enabled the model to correctly predict that the above utterance was `Inclusive Language` rather than the previous erroneous prediction of `Neutral`.

The default ZOOM Automatic Speech Recognition (ASR) tool used in this research has an estimated word error rate of 22%. Future ASR improvements will lead to more accurate transcriptions and consequently, further improve performance of the model.

Incomplete Sentences

Some of the misclassified utterances were incomplete sentences (sentence fragments). Incomplete sentences usually do not have enough information to be *inclusive* or *exclusive* and were usually labeled as `Neutral` by the annotators. This accounted for about 25% of errors during our analysis.

Example: You can

Other Errors

The above errors are not mutually exclusive and, hence, cover less than 90% of errors made by our model. The other over 10% of errors had no discernible pattern.

Part 2: Pointwise Mutual Information and System Errors

The second part of error analysis was a computational linguistic analysis where we used Pointwise Mutual Information (PMI) to discover the most discriminative ngrams, POS and dependency tags that could be included as features for the classifiers. In the context of this analysis, we used PMI as a measure of correlation between an ngram, POS tag or dependency tag and a class label to discover the most discriminative ngrams and tags of a class. The following linguistic features were discovered through the analysis and had an absolute PMI value greater than 0.5:

- An utterance that contains a *nominal subject* is positively correlated with *inclusive language* and negatively correlated with *exclusive language*
- An utterance that contains a *direct object* or

[1] for example, regular expressions similar to the following could be used to detect and remediate many of these speech disfluencies: \b(\w+(?:\s*\w*))\s+\1\b

open clausal complement is positively correlated with *exclusive language*

- An utterance that contains a *VB* (*1st/2nd-person, singular/plural, present-tense verb*) is positively correlated with *exclusive language* and negatively correlated with *neutral language*

Furthermore, as part of our error analysis, we explore bi-grams and trigrams beyond the common pronouns that often imply inclusivity (we, our, us) or exclusivity (I, me, my, you, your). Hence, we do not report (ngram, label) combinations where that ngram includes a pronoun consistent with its commonly assumed language type label. For example, since the bigram *we are* contains the pronoun 'we'–a pronoun that typically implies inclusivity– we do not report the high PMI between *we are* and Inclusive language. In contrast, we do report the high PMI between the bigram *you think* and Inclusive language, since the pronoun "you" is, otherwise, typically associated with *exclusive language*. Equation (1) is the formula we used to obtain the ngrams. Results for the top 3 PMI values for each label are shown in Table 6. The top-3 ngrams were used as features in a revised model, discussed in the next section.

$$\forall\, ngram \in Dataset: pmi\,(ngram, Label) = log\frac{p(ngram \mid Label)}{p(ngram)}$$
$$where\ n = \{2,3\}\ and\ Language\ Type = \{Inclusive, Exclusive\} \quad (1)$$

ngram	Label	PMI	Example
how do	`Inclusive`	1.65	*How do* I do that
want to	`Inclusive`	0.81	Do you *want to* try to do it
to do	`Inclusive`	0.79	Do you want *to do* the next one
run it	`Exclusive`	2.21	just *run it* with this only
just do	`Exclusive`	1.70	So just, *just do* it
back to	`Exclusive`	1.39	Go *back to* that function again

Table 6: Results of PMI Analysis on Ngrams

Results From Augmenting BERT With Linguistic Features Obtained From Error Analysis

We included our findings from part 1 of our error analysis as a data preprocessing step. Speech disfluencies and incorrect cases were fixed prior to the revised model training and testing. To help the model handle incomplete sentences, we include a categorical feature that checks if an utterance has both a verb and subject using the part-of-speech and dependency tags. To augment BERT with linguistic features discovered in part 2 of our error analysis, we concatenated associated categorical features with the embeddings of the transformer before the classification layer. The hyperparameters were the same as the previous model. The resulting model obtained from fine-tuning BERT with linguistic features improved the macro-average F_1-score for detecting `Exclusive Language` by 8.5% to 0.51 while maintaining the already high F_1-score of 0.97 for detecting `Inclusive Language`.

Precision-Recall Trade Off

The precision-recall tradeoff is premised on the fact that an increase in precision of a ML model will typically lead to a decrease in the recall of the model, and vice-versa. Since the end goal of this model is to detect *exclusive language* and occasionally nudge students to help improve their communication, achieving high precision detection is more important than recall. This is because, if the system classifies an utterance as *exclusive language* in error, the resultant system feedback could be counterproductive. Also, in practice, it is not necessary to detect every instance of *exclusive language*–it is enough to provide occasional corrective nudges to the students. Consequently, to assess the possibility of higher precision detection, we introduced the use of a confidence score of 90%. That is, we assume a downstream system will only consider a nudge if our model's prediction confidence is greater than 90%. We use the normalized soft-max probabilities as the confidence score. These are obtained by applying a softmax function to the raw logits output by the transformer. More formally, $\sigma(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{3} e^{z_i}}; i \in \{1,2,3\}\ and\ the\ logits\ \boldsymbol{z} = (z_1, z_2, z_3) \in \mathbb{R}^3$ where z_i is the unnormalized probability of class i (Inclusive, Exclusive, Neutral)

The only distinction between this model and the BERT + Linguistic Features model is specifying that the confidence (softmax) score must be greater than 90%. All other hyperparameters are the same as the revised model. Note, the testing dataset had at least three utterances classified as Exclusive language with over 90% confidence for each student in each of the three homework assignments. Hence, a downstream system using this data would have been able to provide each student with ample appropriate nudges. The resulting model had a macro-average precision of 1.0 for detecting Inclusive language and 0.70 for detecting Exclusive language.

Discussion

Our baseline and inter-annotator F_1-scores show that the task of detecting *exclusive language* is not trivial and the error analysis highlights the context-sensitive nature of the task.

The very high recall values achieved by both human annotators indicates that our gold standard dataset incorporates nearly all instances of *inclusive* and *exclusive language*. While the focus of this paper is on detecting *exclusive language*, our models' ability to also detect the appropriate use of *inclusive language* is beneficial and can potentially positively reinforce desired collaboration decorum. The performance of our models on both tasks is very promising. The results show that the revised model for detecting `Exclusive Language` not only significantly outperformed the baseline on all metrics but also had as high a precision as the human annotators in detecting both `Inclusive` and `Exclusive Language`. Also, it is very reassuring that the model did not misclassify *exclusive language* for *inclusive language* or vice versa. Hence, our model would rarely (in this data, never) nudge students to use *inclusive language* at a time when they are already doing so, which could be counterproductive. However, there is clear room for improvement in the area of recall, particularly for *exclusive language*. In that regard, to encourage further work in this area, we release our dataset for other researchers to use[2].

Some of the linguistic features obtained from part 2 of our error analysis enabled a 13% relative improvement in the precision of our BERT model. We used spaCy[3] to obtain the dependency parses and part-of-speech tags of the utterances which enabled us to analyze and discover potentially relevant linguistic features. An ablation study revealed that the part-of-speech and dependence parse-based features obtained from the computational linguistic error analysis improved the performance of the revised model, though the ngram-based features did not lead to model improvements. This is likely because the fine-tuned BERT model already encodes the ngrams. However, the ngrams obtained from the computational linguistic error analysis reveals that beyond the common pronouns, many interrogative utterances imply inclusivity while many imperative utterances imply exclusivity during collaboration. This is consistent with work done by Cocharan and Sabella (2008) which showed that during collaboration, questions used for clarification, offering an (alternative) idea, or to guide peer understanding foster effective collaboration. The superiority of interrogative-driven over imperative-driven collaboration was further shown in interviews conducted by Cocharan and Sabella (2008) where students stated their preference for and the benefits of question-driven collaborative approach, one of which is inclusive participation. To this end, future researchers can explore including features such as whether an utterance is an imperative or interrogative with the aim of improving the model.

Also noteworthy is the practical emphasis on the precision of our model–a high precision in detecting *exclusive language* is more important than a high recall. This stems from the fact that feedback from a false positive case of *exclusive language* would be counterproductive and we do not have to provide corrective nudges after every instance of such language. To accomplish this, we used the softmax probabilities as the confidence scores which improved the precision of our models–we achieved a 15% relative improvement in precision over the augmented BERT model. However, the confidence scores of Neural Networks are not well calibrated. Guo et al. (2017) explored some Neural Network confidence score calibration techniques which include binning methods, vector and temperature scaling. Future researchers can explore some of these calibration techniques to improve the confidence scores and consequently the precision of the model.

Finally, since we establish data independence by splitting the data both by student and homework to guarantee the models did not overfit to the idiosyncrasies of an individual homework or student, the results published in this paper should be typical of what should be expected with new students and assignments in RPP environments.

Conclusion and Future Work

This paper presents the first dataset of remote pair programming dialogue annotated to indicate utterances with exclusive language. The corpus is of very high quality, estimated to identify over 99% of the relevant utterances, and will be released to enable other researchers to advance this and related tasks. Furthermore, we have proposed a model for automatically detecting the use of exclusive language which could, for example, be used in CITS to improve students' collaboration by detecting the use of exclusive language and nudging the students to use more inclusive language during collaboration. Our classifier, based on fine-tuning the BERT model, significantly surpassed the baseline model and was more precise than the human annotators in detecting the use of inclusive and exclusive language during collaboration.

Our error analysis on the instances misclassified by our best model in the tuning dataset revealed features that improved the performance of our model. Additionally, the analysis highlights the importance of question-based interaction in fostering inclusive collaboration. In future, we will deploy and evaluate our models in a CITS to determine the potential of our model to foster improved collaboration by encouraging inclusive language.

[2] https://doi.org/10.6084/m9.figshare.20493324.v1

[3] https://spacy.io/usage/linguistic-features

References

Begel, A.; and Nagappan, N. 2008. Pair programming: What's in it for me? ESEM'08: Proceedings of the 2008 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, October 2008,120–128 https://doi.org/10.1145/1414004.1414026

Casamayor, A.; Amandi, A.; and Campo, M. 2009. Intelligent assistance for teachers in collaborative e-learning environments. Computers & Education, 53(4), 1147-1154.

Cochran, G. L.; and Sabella, M. S. 2008. Understanding and encouraging effective collaboration in introductory physics courses. In AIP Conference Proceedings (Vol. 1064, No. 1, pp. 95-98). American Institute of Physics.

Forslund F. K.; and Hammar C. E. 2018. Student collaboration in group work: Inclusion as participation. International journal of disability, development and education, 65(2), 183-198.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On calibration of modern neural networks. In Proceedings ofthe International Conference on Machine Learning (pp. 1321-1330). PMLR.

Hughes, J.; Walshe, A.; Law, B.; and Murphy, B. 2020. Remote pair programming. CSEDU 2020 - Proceedings of the 12th International Conference on Computer Supported Education, (Volume 2), 476–483. https://doi.org/10.5220/0009582904760483

Jeong, H.; and Hmelo-Silver, C. E. 2016. Seven affordances of computer-supported collaborative learning: How to support collaborative learning? How can technologies help?. Educational Psychologist, 51(2), 247-265.

Jones, K. S. 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of documentation.

Landis, J.R.; and Koch, G.G. 1977. "The measurement of observer agreement for categorical data". Biometrics. 33 (1): 159–174. doi:10.2307/2529310. JSTOR 2529310. PMID 843571.

Michailidis, N.; Kapravelos, E.; and Tsiatsos, T. 2018. Interaction analysis for supporting students' self-regulation during blog-based CSCL activities. Journal of Educational Technology & Society, 21(1), 37-47.

Oyeleye, L.; and Ayodele, A. 2012. Interrogative utterances as discursive strategy in legislative interactional discourse. International Journal of English Linguistics, 2(5), 122-130.

Pantelides, K.; and Bartesaghi, M. 2012. So what are "we" working on? Pronouns as a Way of Re-Examining Composing. Composition Studies, 24-38.

Pennycook A. 1994. The politics of pronouns, *ELT Journal*, Vol- ume 48, Issue 2, April 1994, Pages 173–178, https://doi.org/10.1093/elt/48.2.173

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; ... and Liu, P. J. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv preprint arXiv:1910.10683.

Tegos, S.; Demetriadis, S.; and Tsiatsos, T. 2014. A configurable conversational agent to trigger students" productive dialogue**:** A pilot study in the CALL domain. International Journal of Artificial Intelligence in Education, 24(1), 62–91. https://doi.org/10.1007/s40593-013-0007-3

Ubani, S.; and Nielsen, R. 2021. Detecting Micromanagement During Pair Programming. In 2021 19th International Conference on Information Technology Based Higher Education and Training (ITHET) (pp. 01-07). IEEE.

Ubani, S.; and Nielsen, R. 2022. Review of Collaborative Intelligent Tutoring Systems (CITS) 2009-2021. In 2022 11th International Conference on Educational and Information Technology (ICEIT) (pp. 67-75). IEEE.

Ubani, S.; and Nielsen, R. 2022. Classifying Different Types of Talk During Collaboration. In International Conference on Artificial Intelligence in Education (pp. 227-230). Springer, Cham.

Ubani, S.; and Nielsen, R. 2021. Detecting Micromanagement During Pair Programming. In 2021 19th International Conference on Information Technology Based Higher Education and Training (ITHET) (pp. 01-07). IEEE.

Ubani, S.; and Nielsen, R. 2022. Review of Collaborative Intelligent Tutoring Systems (CITS) 2009-2021. In 2022 11th International Conference on Educational and Information Technology (ICEIT) (pp. 67-75). IEEE.

Ubani, S.; and Nielsen, R. 2022. Classifying Different Types of Talk During Collaboration. In International Conference on Artificial Intelligence in Education (pp. 227-230). Springer, Cham.