A Low-code Approach to Identify Toxicity in MOBA Games

Alexandre Vaz
Eng. de Sist. e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
adonnellyvaz@cos.ufrj.br

Alexandre Batista
Eng. de Sist. e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
absilva@cos.ufrj.br

Farmy Silva
Eng. de Sist. e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
farmygfs@cos.ufrj.br

Lincoln Magalhães Costa
Eng. de Sist. e Computação, COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
costa@cos.ufrj.br

Geraldo Xexéo
Eng. de Sist. e Computação, COPPE
Dep. de Ciência da Computação, IM
Universidade Federal do Rio de Janeiro
Rio de Janeiro, Brasil
xexeo@cos.ufrj.br

***Abstract*—In this work, we explore the use of KNIME to identify toxic behavior in the MOBA game DOTA 2. Using a dataset composed of 10530 messages taken from 1903 matches, we tested the use of KNIME to identify toxic messages obtaining an accuracy of 92% and 85% for toxic and non-toxic messages, respectively. The DOTA 2 game chat log was used to present a low-code approach to a supervised learning model for message classification. In addition to providing insight into the toxic behavior of MOBA players, our work supports the idea that low-code development can reach levels as good as traditional development. On the other hand, our study can also serve as a basis for more elaborate implementations that allow us to observe other aspects of toxic behavior from its detection, encouraging the construction of prevention and neutralization tools.**

***Index Terms*—Toxicity in Games, Low-code, Natural Language Processing, DOTA 2**

I. Introduction

Online digital games usually offer communication channels between its players, however, this communication is often of an abusive nature. Abusive behavior is identified through the recurrent use of aggressive, hateful, and offensive speech by some players when talking to others. This is referred to as the toxicity of communication systems in digital games. In an attempt to solve this ongoing problem, the gaming industry relies on match reporting systems that rely on expert group analysis, or automated systems capable of detecting inappropriate messages. The absence of these efforts to moderate in-game communication can lead to an environment in which certain players may feel hurt or intimidated and, consequently, abandon or avoid playing the game, even after winning a match.

The social aspect of games is at times considered to be an attraction, as communication is an instinct used in game design intended to cause players to feel joy and excitement and give them a greater sense of identification [1]. On the other hand, online in-game communication can also be a burden to the gaming community, as reports of misuse of communication are increasing [2]–[6]. The challenge in dealing with this problem is that messages often contain misspelled and abbreviated words, unstructured sentences, slang, game-related terminology, and out-of-context phrases; all of these factors add stochastic noise to essential pattern detection for machine learning algorithms.

This study was motivated by the challenge of presenting an approach to identifying toxic messages in an online game using a low-code tool. Defense of the Ancients 2 (DOTA 2) was chosen for this study because it is a game with highly competitive characteristics, where two teams are divided into an equal number of players. Communication occurs between members of a given team and in parallel between all competitors in a match.

We corroborate with some works that already use methods based on machine learning, and that seek to make machine learning more accessible and easier to apply (Section II-B). This work differs too much in terms of presenting the use of a fully visual tool, based on drag and drop, providing simple mechanisms for the quick resolution of a problem (Section II-A). The main contribution of this work is the demonstration that a low code solution can achieve high levels of effectiveness in classifying toxic messages (Section III). Discussions are presented in Section IV. And it concludes by presenting a list of approaches that can be developed based on this study (Section V).

II. Background and Related Work

The literature on low-code solutions shows some approaches that make artificial intelligence more accessible and easier to apply [7], [8].

A. KNIME As a Low-Code Tool

Visual programming has become quite popular in recent times and aims to replace, partially or completely, the practice of traditional coding. In addition to making it easier and more accessible. In visual programming, a graphical user interface (GUI) guides you through all the steps necessary to create a pipeline (workflow) of dedicated blocks (nodes).

Thus, each node implements a particular task; each node workflow carries its data from the beginning to the end of its flow. A workflow replaces a script; a node replaces one or more lines of script.

In KNIME, nodes are created by dragging and dropping (or double-clicking) from the Node Repository to the Workflow Editor (Workflows), which is in the center of the workspace. So, node after node, the pipeline is quickly built, configured, executed, inspected, and documented.

Figure 1 shows a basic KNIME workflow presented in their website where the data is read, filtered by column and row, and in the end two charts are presented.

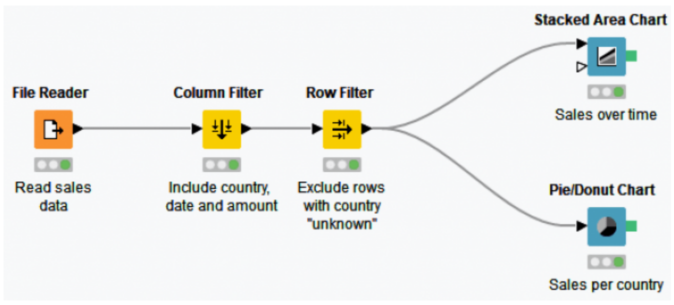

Fig. 1. KNIME basic workflow. Source: KNIME Website

B. Related Works

Several proposals for the identification of toxic behavior in online games exist [2], [5], [6], but most are related to more complex approaches to development approaches and in some cases to deeper knowledge about the functioning of artificial intelligence and its approaches. This work found only three works that address the topic of identification of toxicity in games using low-code tools and that also bring a certain complexity.

Martens et al. [4] use data obtained through an extinct DotA platform called *DotAlicious* to classify toxicity messages sent in-game. Several interesting choices can be pointed out from their methodology. Among them is the use of a *letter set* class, to detect variations of words with extra letter repetitions, for example, if a player writes "noooob" instead of its original form "noob". Another one is the toxicity classification: n-grams (for n=1,2,3,4) within messages that contained "bad" words were classified for players that had played at least 10 matches.

Maiya [7] presents ktrain[1], a low-code Python library that makes machine learning more accessible and easier to apply. As a wrapper to TensorFlow and many other libraries (e.g., transformers, scikit-learn, stellargraph), it is designed to make sophisticated, state-of-the-art machine learning models that are simple to build, train, inspect, and apply for a wide range of users, from beginners to experienced practitioners.

Another low-code solution is presented by Gwendal Daniel et al [8], which introduces a framework called Xatkit. The framework addresses the problem through a set of chatbots (voicebots and bots in general) defined by Domain-Specific Languages in a platform-independent way.

III. Experiment

A. Data Collection

DOTA 2 provides an open source application programming interface (API) called OpenDota [9]. Amateur DOTA 2 players have the option to choose to have match data collected via the API, and all competitive and professional matches are automatically part of OpenDota. While direct access via API requests is a great choice for experiments dependent on large amounts of data, this article opts for an aptly named dataset *DOTA 2 Match Dataset* [10] put together by Joe Ramir and available on the Kaggle platform [11].

B. Authors' Expertise

The authors' level of familiarity with DOTA 2 varies greatly. And this difference affected the toxicity rating, as it involved a learning curve regarding specific terms of the gaming community. An example is the term "creep", usually attributed to someone who exhibits unpleasant behavior [2] and therefore has a negative connotation, but in DOTA is a game component with a neutral connotation. This example demonstrates how terminology unique to online gaming communities can influence the perception of toxicity within their context.

C. Data Filtering, Cleansing, and Classification

A significant portion of the messages was not classifiable, either because they were written in languages that the outside of the authors' domain or they were exclusively written in symbols i.e., numbers, special characters, among others. Therefore, the following criteria were established for the data classification, only messages written in English, Portuguese, and Spanish were classified; also, messages that only included *emoji*, numbers, symbols, were ignored by the classification. A total of 1750 messages originally present in the dataset did not meet the criteria and were not included in the experiment. The remaining 8780 messages, pertaining to a total of 1903 matches, were distributed among the specialists for a simple, binary classification, where messages identified as toxic were given a value of (1) and otherwise given a (0).

D. Learning and Classification Model

After undergoing the process of filtering, cleansing, and manual classification described in the previous section, the data served as an entry to this study's learning model, which uses the low-code program KNIME and the Palladian toolkit to facilitate the nodes structuring. The process begins by reading the data and filtering out the data manually indicated

[1] github.com/amaiya/ktrain

[2] https://dictionary.cambridge.org/us/dictionary/english/creep

as unclassifiable and goes on to a node that partitions the data and performs cross-validation. The cross-validation works by iterating through a loop, in which each iteration is designated to one of the data partitions [12]. The partitioner node also sets aside part of the data for learning, using the *Text Classifier Learning* node, which receives the manually pre-classified data as a parameter and utilizes a table with weights assigned for each term to determine how probable each feature is to each category [13]. Subsequently, the *Text Classifier Predictor* receives the model built by the *Text Classifier Learner* and the test data partitioned by the *X-Partitioner* to classify the messages' toxicity and measure the models' efficacy. Lastly, the data goes through an *X-Aggregator* node, responsible for the calculation of the model's data as a whole [14] and passes it on to the *Scorer* node; a node that compares two columns by their attribute value and displays the confusion matrix, which is the number of lines that correspond correctly to their classification. The *Scorer* node also returns other statistics such as: true positives, false positives, true negatives, false negatives, information retrieval, precision, sensibility, specificity, F-score, overall precision, and the Kappa coefficient [15].

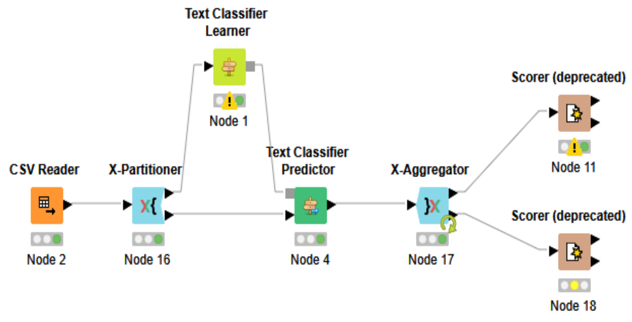

Fig. 2. The KNIME *pipeline* beiong used as as learning model to classify toxic messages.

E. Results

The findings obtained by this preliminary study according to the experiment detailed in the previous subsections are displayed in the following tables.

Table I presents the experiment's confusion matrix, in which a one (1) indicates a message as toxic and a zero (0) signifies that a message is non-toxic. On the left, 0 and 1 represent the real classification given to the message, and the table's first line indicates how the model classified them. The model reached a mean accuracy of 87% in this context. The results are also satisfactory when analyzing the errors, since false negatives are more important than false positives when dealing with the classification of toxic messages, and only 246 messages were erroneously treated as non-toxic.

Meanwhile, the advanced classification metrics like revocation, precision, sensibility, and F-score are available in Table II. The Kappa coefficient was used to measure the experiment's reliability and had a score of 0.75, being therefore considered *substantial* [16].

TABLE I
CONFUSION MATRIX

		Predicted	
		0	1
Real	0	3205	776
	1	247	4410

TABLE II
ADVANCED METRICS

	Revocation	Precision	Sensibility	Specificity	F-Score
0	0.805	0.925	0.805	0.945	0.861
1	0.946	0.846	0.947	0.801	0.894

IV. DISCUSSION

Precision is defined as the fraction of correctly classified items among the total number of items retrieved by a query [17], for example, the percentage of truly non-toxic messages when retrieving the non-toxic messages in the experiment's database. Revocation is defined as the percentage of the documents relevant to a query that is actually retrieved [17]. For example, if 100For example, if 100 documents are relevant but only 30 are retrieved by the query, then it is attributed a revocation of 0.3.

documents are relevant but only 30 are retrieved by the query, then it is attributed a revocation of 0.3. As seen in the results, the experiment's revocation was consistently above 0.8. The revocation of the toxic messages stands out, with 95% of the results being retrieved. The F-score, a harmonic weighted average of the precision and revocation scores using a constant β to determine the weight given to each score, can also be used to measure an information retrieval model's performance [18]. The β chosen for this experiment's F-score was of one (1), such that revocation and precision were given an equal weight [18], resulting in a relatively high F-score for both the toxic and non-toxic messages, receiving 0.84 and 0.89, respectively. A possible explanation for the model's success in classifying toxic messages is the repeated use of tokens that were always attributed as toxic, including racist and homophobic slurs, along with toxic game-related slang like the term "ez".

V. CONCLUSION AND FUTURE STUDIES

A possible improvement to this experiment would be to differentiate between neutral and positive messages or to have a scale of how toxic a message is, rather than using a simple, binary classification. A learning model with more outputs would be necessary to accommodate the change from a binary to a ranking with multiple levels of toxicity. The following labels could be given to each level: positive, vaguely positive, neutral, vaguely toxic, toxic, and hate speech. This change to results of greater granularity would most likely reduce the model's accuracy, however, it would be more informative of nice, positive messages and differentiate toxic messages from intolerable hate speech.

DOTA 2 is a competitive game and provides two communication channels with the following objectives: a) General: dedicated to the communication of the match as a whole, so that information is exchanged about the game. b) Team: exclusive to team members, useful to discuss tactics and other information to gain an advantage over the opposition.

Despite the goals set for each of the channels, players give it another usefulness and meaning. The general chat becomes a space to humiliate and insult other players, when they are losing, for example. Meanwhile, team chat is used to offend other team members. Gaming toxicity occurs when a message of negative connotation and derogatory content is sent, interrupting expected sporting conduct and, at times, the behavior includes, but is not limited to, racism, xenophobia, homophobia, and other unacceptable acts of hate in any segment of society. It is also important to note that there are toxic messages from the gaming community, such as the "ez", which means that the victory over an opponent was easy.

An accuracy of 92 % and 85 % for toxic and non-toxic messages, respectively, using a total of 8,784 pre-sorted messages, shows that it is possible to determine the toxicity of messages in MOBA games. This positive result allows the use of the database and the learning model of this study in subsequent experiments related to toxicity classification in in-game chats. The results are also satisfactory in error analysis, as false negatives are more important than false positives when it comes to classifying toxic messages. While framing non-toxic players as toxic players might seem like a mistake, a rating system is not a punishment system. In other words, taking the message for analysis is more favorable than letting a toxic message persist.

The attributes provided by the OpenDota API requests, some of which were used in this study, cover a large number of possible areas and topics of research related to MOBA games and games in general. Some of the possibilities that could branch out into this study using other attributes publicly provided by the OpenDota API are, but are not limited to: I) Using the player's usernames permits the identification of players that emit toxicity during the course of a game, enabling a study investigating the correlation with the player's toxicity and the team's results; II) While the item above deals with overall team results, OpenDota also contains information about individual player statistics, so a correlation between player's kills, assists, deaths, and so on and their message's toxicity could be analyzed; III) A sentiment analysis of the messages exchanged during the match is an alternative to classifying their toxicity; IV) Since the OpenDota API is used in professional matches, regional leagues, and casual gameplay, one could study the varying levels of toxicity among these different contexts.

This study demonstrates that low code tools can be effective in toxic message classification tasks. And the use of Knime demonstrates that low-code development can reach levels as good as traditional development. Making this study a basis for more elaborate implementations that allow us to observe other aspects of toxic behavior.

Related studies [4], [7], [8] present approaches that depend on a broader level of knowledge about artificial intelligence and programming. This work is not as dependent on knowledge of artificial intelligence, but demonstrates the abstraction of deeper knowledge in development. This allows for a series of actions that can gradually increase the potential market of the game by making it a more welcoming environment.

Acknowledgment

This study was financed by the National Council for Scientific and Technological Development (CNPq) - Brazil under the Concession NORMATIVE RESOLUTION RN-017/2006; and the Coordination for the Perfection of Higher Education Personnel (CAPES) - Brazil under the Financial Code of the Scholarship 001. Farmy Silva was supported by CNPq

References

[1] R. Dillon, *On the Way to Fun: an emotion-based approach to successful game design*. CRC Press, 2010.

[2] J. Blackburn and H. Kwak, "Stfu noob! predicting crowdsourced decisions on toxic behavior in online games," in *Proceedings of the 23rd international conference on World wide web*, 2014, pp. 877–888.

[3] H. Kwak and J. Blackburn, "Linguistic analysis of toxic behavior in an online video game," in *International Conference on Social Informatics*. Springer, 2014, pp. 209–217.

[4] M. Märtens, S. Shen, A. Iosup, and F. Kuipers, "Toxicity detection in multiplayer online games," in *Proceedings of the 2015 International Workshop on Network and Systems Support for Games*, ser. NetGames '15. IEEE Press, 2015.

[5] Y. Kou, "Toxic behaviors in team-based competitive gaming: The case of league of legends," in *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, 2020, pp. 81–92.

[6] N. A. Beres, J. Frommel, E. Reid, R. L. Mandryk, and M. Klarkowski, "Don't you know that you're toxic: Normalization of toxicity in online gaming," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–15.

[7] A. S. Maiya, "ktrain: A low-code library for augmented machine learning," *arXiv preprint arXiv:2004.10703*, 2020.

[8] G. Daniel, J. Cabot, L. Deruelle, and M. Derras, "Xatkit: A multimodal low-code chatbot development framework," *IEEE Access*, vol. 8, pp. 15 332–15 346, 2020.

[9] Opendota api (v18.0.0). [Online]. Available: https://docs.opendota.com/

[10] Dota 2 matches dataset: Base de dados. [Online]. Available: https://www.kaggle.com/jraramirez/dota-2-matches-dataset

[11] kaggle, comunidade on-line de cientistas de dados. [Online]. Available: ‘https://www.kaggle.com/

[12] K. partitioner. (2020) X-partitioner. [Online]. Available: https://hub.knime.com/knime/extensions/org.knime.features.base/latest/\org.knime.base.node.meta.xvalidation.XValidatePartitionerFactory

[13] K. T. Classifier. (2020) Text classifier. [Online]. Available: https://www.knime.com/book/text-classifier

[14] K. aggregator. (2020) X-aggregator. [Online]. Available: https://hub.knime.com/knime/extensions/org.knime.features.base/latest/\org.knime.base.node.meta.xvalidation.AggregateOutputNodeFactory

[15] W. Tang, J. Hu, H. Zhang, P. Wu, and H. He, "Kappa coefficient: a popular measure of rater agreement," *Shanghai archives of psychiatry*, vol. 27, no. 1, pp. 62–67, Feb 2015, 25852260[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25852260

[16] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012, 23092060[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/23092060

[17] O. N. P. Cardoso, "Recuperação de informação." *INFOCOMP Journal of Computer Science*, vol. 2, no. 1, pp. 33–38, 2000.

[18] P. F. Matos, L. Lombardi, R. Ciferri, T. Pardo, C. Ciferri, and M. Vieira, "Relatório técnico "métricas de avaliaçao"," Universidade Federal de Sao Carlos, Tech. Rep., 2009.