

# Pavement Defects Detection Based on Yolov5 and Generative Adversarial Network

1<sup>st</sup> Amara Tariq  
*Department of Computer Science*  
*Boise State University*  
Boise, Idaho, United States  
amaratariq@u.boisestate.edu

2<sup>nd</sup> Hongmin Kim  
*Department of Computer Science*  
*Boise State University*  
Boise, Idaho, United States  
stevenkim@u.boisestate.edu

**Abstract**—The world of gaming has grown and evolved tremendously, especially as the world went fully remote and was based online for the duration of the COVID pandemic. Due to this increase in the number of players, the levels of toxic gameplay are growing, which deters new players from the game. In this paper, we designed a software that checks in-game chats to be able to determine who the toxic players are; predicting who is most likely to be reported. We leverage sentiment analysis in order to understand the tone of the messages and the intent behind them. Evaluation of our tool was done by testing it against a dataset containing chat logs from multiple games off League of Legends games with labels of the reported and banned players. Our software would help streamline games checking to make sure the report is valid and whether that player should be banned, as well as having potential to warn players regarding their behavior prior to passing the threshold where they get banned.

## I. INTRODUCTION

Dissociative anonymity is a growing concern in the community surrounding video games. So much so that the platform Unity did a new study on online behavior and found that 70% of players have experienced some form of toxic behavior [1]. Playing video games fundamentally involves constant communication and interaction with teammates, and some games even allow communication with the enemies. In order to succeed, players have to consistently communicate and work with their teammates in order to gain experience and rank up their accounts to move up in the game.

Unity conducted another study in 2021 focusing specifically on the Toxicity in Multiplayer Games [2]. They found that even more players, about 72% of players have witnessed toxic behavior, while 68% of them actually have experienced it themselves. Of this player base, they found that 43% of the players had actually completely quit the games that they play. With almost half of the player base actually deciding to step away and stop playing the game, it becomes important for the game companies themselves to try and change the environment to allow their players to continue wanting to play. More than 3 in 4 multiplayer gamers agreed that protecting players from toxic behavior should be a priority for game developers [2]. Without any change, not only can the level of toxicity potentially grow, but more and more players are going to reach their thresholds on the amount of toxicity they can handle, which will cause them to quit.

Most multiplayer games already use a reporting system that allows the participants in the game to report their teammates, and even enemies, for various reasons. Toxicity is one of the various reasons to report someone, but sometimes these reports are made out of rage. “Gamer rage” is a term simply used for when people get really angry from the events occurring in-game, which can then lead to the player reporting people just because they might have done something the player did not like. This further creates an environment where players feel unsafe, especially those who might be new to the game. Sometimes players will even convince the team to all report the same person, in an attempt to get them banned. This is called “mass reporting” and is a problem that developers need to also address. [3]

In this paper, we describe some tools to be able to read in-game communication and predict who is the player that will get reported. We then analyze our findings to see if toxicity played a role in the types of messages the reported player sent. We then explore whether there is a correlation and if game developers would benefit from using these tools for the game chat in real-time.

## II. BACKGROUND

Although there have been studies and research conducted in this area, not many companies or games have solutions to solving this problem. A lot of the growth in toxicity occurred over the COVID-19 pandemic, which made it hard for companies to react to as various countries all over the globe went into lockdowns. Since the decline in cases and ease of restrictions, a group of game companies formed the Fair Play Alliance, a coalition working to address harassment and discrimination in the gaming industry [4].

Many companies are slowly starting to intervene and ban players that show toxic behaviors. Such as the live-streaming platform Twitch, who has been actively banning several prominent streamers for reasons from sexual harassment to racist and discriminatory language in game [4]. In April 2022, Riot Games decided to permanently ban a player due to their misogynistic remarks, with later clips showing them threatening rape and sexualizing underaged girls. It took the internet spreading these clips for companies to be made aware and take action. These are the types of conversations that

companies need to safeguard their player base from, all whilst also being able to effectively warn and potentially remove the toxic player all together. This is a start, but companies also have no guarantee that the player won't make a secondary account and continue to access and play the games, all while continuing their previously toxic behavior.

In July of 2018, companies like Ubisoft activated chat filters that would filter words that were deemed inappropriate, banning any players that used those words [4]. Although it did take some action and ban players, many found that the filters were frustrating and ruined communication in game. Players showed that even the message "hello" would get censored to "\*\*\*\*o". This showed how precise the censor filters had to be; just enough to be able to censor inappropriate language, yet lenient enough to allow players to continue to chat amongst one another. Knowing the precision that was required in order to read in only the inappropriate words, the model and sentiment analysis that we performed needed adjustments. Due to this, custom libraries needed to be made for words that would end up having a different connotation in game versus the dictionary and usage in the real world. For example, if a person were to use the phrase "I got 2 kills," the usage of "kill" would be seen as negative, but in-game the same phrase would be seen as neutral, if not almost positive. When games include chat filters, not only do words need to be checked, but also the context in which it was used. Currently there are not any chat filters that support that type of feature.

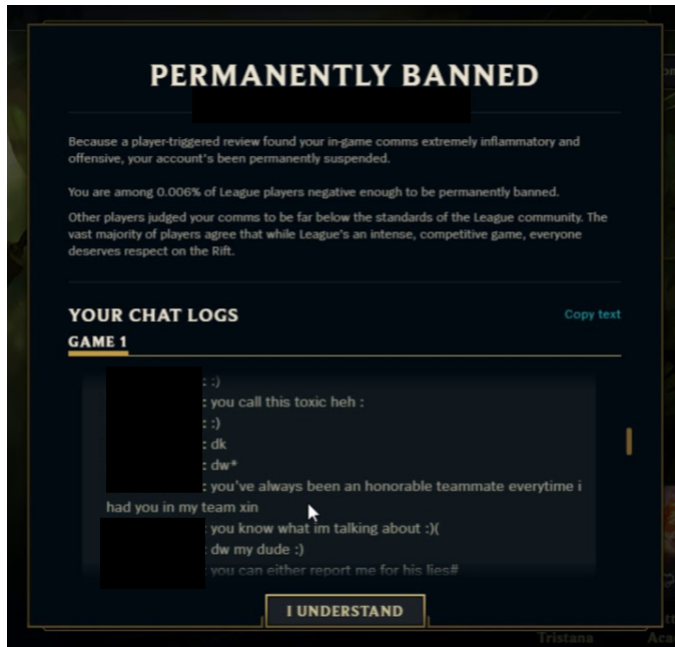


Fig. 1. What a player sees when getting ban

Most games do not tell players that they have been reported unless it gets to the point of the player getting banned. This can be seen for League of Legends in Fig. 1. With this information being shared so late, it allows players to continue toxic behavior for the duration of not only an entire game, but

up until the player actually has disciplinary actions applied on them. During this entire time period, multiple people and other users would be interacting with them, and potentially even quitting the game because of the interaction. Riot Games has tried to implement an "instant feedback report" that would issue a message that can be seen below in Fig. 3. This feedback is a very generalized message and states that a "penalty" was issued, but there are no actions that are taken, for example a temporary ban from launching into a new game could be a penalty.



Fig. 2. What a player sees when getting ban

In 2014, League of Legends did try and be proactive in regards to their growing player base becoming more toxic. The program was named The Tribunal and was quickly disabled and shutdown indefinitely. Riot Games never commented as to what caused this removal, so players can only speculate.

This all leads to the reason this is important. If games could enhance this process by simply adding warnings to the player every so often or once they get close to the threshold of being banned, it could help to curb toxicity and let players have a chance to take a break and change how they chat in-game. This could be done by applying a model or sentiment analysis that can see trends in the chat and find the toxicity level at which the probability of the player getting reported is high. Then a system could be created to warn players before they actually get reported or banned would help keep them in check. Games could be kept more friendly and allow more people to feel comfortable and safe.

### III. DATA

The game we chose to focus on was Riot Games' League of Legends. The game is known to be toxic, according to a study done by the ADL, an anti-hate organization whose goal is to fight hate [6]. Not only was the game proven to have a toxic environment, but having had familiarity with the game is also necessary. The experience with the game allows better understanding of terminology specific to the game as well as knowledge as to what different events may cause players to want to report others. This can be done both due to chat reasons, but also actual in game plays and situations, all which need to be accounted for.

The dataset that was used covers actual games played, with several different European languages being included. The dataset had chat logs, with each chatlog being the chat from a single game, which spanned around 10,000 total games. [7]

In order to create the most effective model, the Python library Spacy's language detector was applied to find the average language spoken for each chatlog. Anything that was not English was filtered out; and then we further filtered out the remaining English chatlogs. The second filter was based on a threshold of 98%, which was set based on the mean of English content in these remaining chatlogs. The resulting dataset contained 8928 chatlogs that were all 98% English or higher.

The labels in the dataset were very useful for what the overall goal was. *Association\_to\_offender* labeled the players from the point of view of the offender, who is the reported player. Based off this tag, the amount of messages was further reduced by the decision to only analyzing the team who had the offender on it, as the enemy team's chat would not provide any relevant data on the reported player.

*Most\_common\_report\_reason* is one of the primary labels for any analysis step. When a player is reporting another, they are given a list of 7 reasons that they can choose from to explain what happen to warrant a report. This form can be seen in Fig. 3. Having this information would help confirm and validate the results the model and sentiment analysis would provide. Cross comparing all of the results would help find a pattern to why players were getting reported. BERT would find a trend and pattern in the actual messages to try and predict the offender. This accuracy is purely dependent on the messages BERT takes in, and not necessarily on toxicity of the messages. This is where the sentiment analysis comes into play.

For sentiment analysis, custom lexicons, or dictionaries, were created for negative, neutral, and positive words. These lexicons had to be altered to account for the differences in terminology and language, as for the reasons discussed earlier. This sentiment analysis would then be applied on each chatlog, with an average score being given to each player. The more negative the value, the more negative and toxic the player potentially was. If the most negative player was indeed the person who got reported, and if the BERT model was able to predict the same players, then it can be said that indeed there is a correlation between the level of toxicity and the reports made against a player. The label *most\_common\_report\_reason* must also be checked. If the tag indicates a reason that relates to gameplay and there was no correlation between the sentiment analysis and BERT model, then it can be concluded that there is another factor that is affecting the reports. If there was no correlation found, but the tag indicates it was indeed toxic behavior, then it can be hypothesized that maybe everyone is toxic; that that is the normal type of chat behavior the game unfortunately sees.

#### IV. MODEL AND APPROACH

The main goal is to create a model to predict who would be the reported player based on the messages, but also check with a sentiment analysis if the reason was actually due to toxicity. If the majority of the model's guesses are due to toxicity of messages, games can then apply this type of algorithm and

Fig. 3. List of reasons a player can be reported for

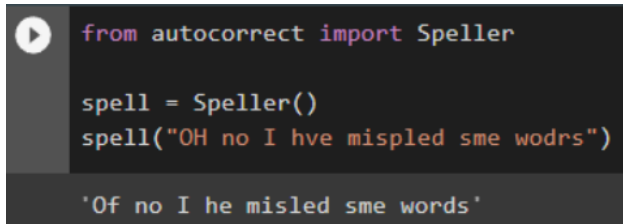
software to help players know when they are being too toxic. Players' awareness of their own behavior could help curb and limit the amount of toxicity in-game, thus allowing more people to enjoy the game.

Therefore, our primary question was to ask whether or not there was a pattern in the chat messages, specifically in the feature of toxicity level, for the players that got reported. Then a secondary question was to verify our predictions by testing against actual League of Legend game records. If these aligned, this means that there is a trend of how toxic a player is in chat and them getting reported.

The first step was to apply text processing steps to the dataset. A limitation that was encountered was that any type of text processing applied to the dataset would cause most of the context and negative words that our process depended on to be lost. Most gaming terminology is slang, words not in an actual dictionary; sometimes the words means something different in the game versus the dictionary definition. Due to this, text processing steps had to be limited. This meant that raw text was used, with no lemmatization or stemming, with no additional steps added.

Python's auto-correct library was also tested, but slang and various game terminology was lost, as this library used a dictionary to check against. The auto-corrector also did not contain methods or algorithms to detect context, thus causing words to auto-correct to other words that did not make sense

with the overall phrase. Simply missing a letter or even having a word in all capital letters would result in a completely different output from the auto-corrector, which is shown in the example in Fig. 4.



```
from autocorrect import Speller

spell = Speller()
spell("OH no I hve mispled sme wodrs")

'Of no I he misled sme words'
```

Fig. 4. List of reasons a player can be reported for

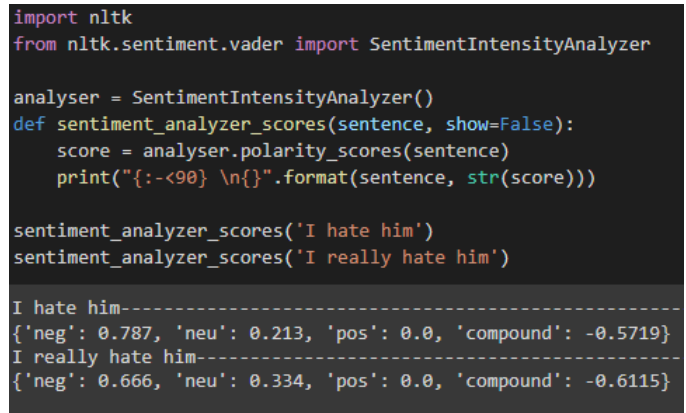
Choosing to not process the text also limited how accurate any model or sentiment analysis could be with misspelled words. There are other methods to find the best fitting word, using the context of the overall sentence, but those methods still use the English dictionary as the foundation to how they correct spelling. For a game chat with so much slang and different terminology, none of these methods could have worked. Thus any lemmatizing, stemming, auto-correct, or any other steps that would alter the spelling of words was avoided due to potentially loosing or changing words that would be necessary for sentiment analysis.

Spacy's language detector was applied here and as described previously, any chatlogs without the majority of chats being in English were filtered. Then the threshold of 98% was created from the mean of the remaining chatlogs. With this step applied, 8928 chatlogs remained with all being 98% English or more.

#### A. Sentiment Analysis

For the sentiment analysis, initially NLTK's Vader was used. Vader is a model that is used for sentiment analysis that not only takes in the polarity of words, i.e. their positive and negative values, but also takes the intensity of the words into account. Fig. 5 shows an example of how Vader not only takes the polarity, but also applies intensity. As the figure shows, the phrase "I hate him" does have a high initial negative rate compared to the second, but the main focus of the output from Vader should be the overall compound score. This score also factors in the intensity, whereas the negative score does not. Thus we can see how overall, just adding a booster word such as "really," the score increased by almost 4%.

Vader also uses a dictionary that then associates each word with its own values. This dictionary was then altered to adjust for the language of these chatlogs as well as for the terminology that usually is used for the game. The terminology was compiled by surveys conducted with Boise State University League of Legends players. Through this process, Vader showed some errors due to the decision made to not auto-correct, as if the word wasn't in Vader's dictionary, it would give a score of neutrality, which is 0.



```
import nltk
from nltk.sentiment.vader import SentimentIntensityAnalyzer

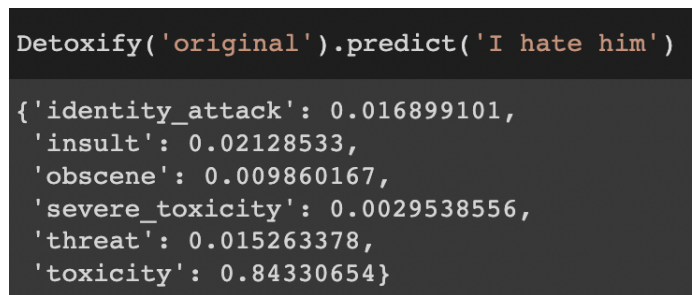
analyser = SentimentIntensityAnalyzer()
def sentiment_analyzer_scores(sentence, show=False):
    score = analyser.polarity_scores(sentence)
    print("{:-<90} \n{}".format(sentence, str(score)))

sentiment_analyzer_scores('I hate him')
sentiment_analyzer_scores('I really hate him')
```

```
I hate him-----
{'neg': 0.787, 'neu': 0.213, 'pos': 0.0, 'compound': -0.5719}
I really hate him-----
{'neg': 0.666, 'neu': 0.334, 'pos': 0.0, 'compound': -0.6115}
```

Fig. 5. Vader Sentiment Analysis Example

The second sentiment analysis tool that was used was Detoxify, which uses the BERT model and Huggingface Transformers. Detoxify uses Kaggle's API to train on its datasets, allowing for it to fine-tune itself. This tool also provides a larger variety of outputs that can be used to analyze with. As seen in Fig. 6, Detoxify analyzes for fields such as threat, obscenity, and even has two classifications of toxicity. For the purposes of this project, we chose to use the general toxicity score as it would give a good general score for the polarity of the messages.



```
Detoxify('original').predict('I hate him')
```

```
{'identity_attack': 0.016899101,
 'insult': 0.02128533,
 'obscene': 0.009860167,
 'severe_toxicity': 0.0029538556,
 'threat': 0.015263378,
 'toxicity': 0.84330654}
```

Fig. 6. Detoxify Sentiment Analysis Example

The reason a second tool was used was to further compare and test our data for a more accurate representation. It was found that, due to Detoxify already being trained and able to detect sentiment even with spelling errors, that was the best tool to use.

#### B. Models

Before applying our sentences through BERT, we tested to see if the TfidfVectorizer would work properly, as a common model that is often used. As explained above, we did not lemmatize nor stem the sentences, which our model would struggle upon finding different word variations of the same root word. Although, if it were to have a higher accuracy, we would be able to tell if there is a relationship with the words and the labels. We designed a pipeline using the

TFIDFVectorizer, a TruncatedSVD with the dimension of 8, and a logistic regression classifier.

Additionally, using both the sentiment analysis values from our custom VADER, and the detoxify library, we made a support vector machine and a logistic regression model, to predict the offenders.

In order to guess the offenders as labels of zeros and ones, we applied BERT, which is a transformer based pre-trained model. Our group decided to use BERT because we have used it in a previous assignment that utilizes BERT, and it acts as a standard transformer model in the area of NLP. In order to use BERT, converting the texts into features, otherwise known as the process of embedding is required. Embedding allows BERT to use the sentence inputs to focus on: word/sentence relationships, word per word prediction of its meanings, and transfer learning to be used as an input of BERT. The BERT model is trained under the following parameters: learning rate = 0.002, epoch = 5, batch = 32, validation split = 0.1. To further assess the differences in the column of *most\_common\_report\_reason*, we decided to pick out the reported reasons of verbal abuse, negative attitude, and offensive language, since they are more related to the chat based reports. From each three separated datasets, we picked the first 10000 rows as the training data, and the next 10000 rows as the test data.

## V. RESULTS AND EVALUATION

The pipeline using TFIDFVectorizer has reached 78% accuracy, where it is the same as the baseline.

Our support vector machine, SVM, and Logistic Regression model each achieved 57% and 59% accuracy, where the baseline is the 57%. This proves that our pipeline and models are not doing anything besides guessing the majority of the classes.

For BERT, we got the following results for the datasets with different reported reasons:

Report reasons	Train Acc	Test Acc	Baseline
Verbal Abuse	72%	73%	67%
Negative Attitude	76%	73%	68%
Offensive Language	74%	73%	67%

Table 1: Showing accuracy and baselines

All three of the datasets reached 73% accuracy, which is quite off from our expectation. It does have a 5% to 6% accuracy rise from the baseline, but it is not performing well to predict who the offender is. We also expected datasets of verbal abuse and offensive language would reach a higher accuracy, yet it does not perform better than the negative attitude dataset. The results indicate that the overall chatting tendency is going to be hard to detect who the offender is and who is not. Moreover, the sentiment analysis and detoxify library showed that there are no correlation with the overall chat and the scores, which further proves our point.

Below is shown our confusion matrix, which indicates all the true positive, true negatives, false negatives, and false positives. As show, we did have a noticeable amount of false

positives and negative. This just further indicates that our model was not predicting anything above the baseline.

		Predicted	
		Yes	No
Actual	Yes	5347	1982
	No	1982	689

Table 2. Confusion matrix of verbal abuse

From this confusion matrix, we can use the equation shown below to calculate the overall accuracy of our model. This accuracy confirms again that the model we have does not compute anything above the base line. This could be due to the specific features that we used, as well as the data itself could lack any patterns. Without patterns the model will be unable to make accurate predictions, thus leaving us with a low accuracy.

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} = \frac{689 + 5347}{689 + 5347 + 1982 + 1982} = 0.6036$$

### A. Limitations

A limitation that occurred in our data was potential false positives in our data. The game data that was used contains just the data on the number of people that reported the player in game, there is no further knowledge in terms of what happen in game. It is unknown if Riot ended up banning the player or if any other factors that influenced the situation. Due to this, an assumption has to be made that there are some false positives in the data. This could be situations where the player that got reported did not actually do anything bad, i.e a new player who happened to be placed in a game with any experienced players. Those experienced players could have potentially become frustrated with a new player and simply reported due to lack of skill. In actuality, that report ended up not being valid, as there was no reason. This limited how accurate our model would be.

Another limitation that may have occurred is the direct limitations of the sentiment analysis. Sarcasm and the tone of chats are difficult to detect. The sentiment analysis that was used tried to detect toxicity, but some of those messages could be sarcastic, which would then alter the scores and therefore could potentially be enough to alter the overall results. Thus we have to keep this fact in mind.

The dataset that was used was also a limitation, as the chatlogs were compiled from games played in 2014. This was during a time that League of Legends held what is known as Tribunals, which was a system they attempted to use to discipline toxic players. The system didn't last long and ended up being disabled indefinitely. As mentioned previously, Riot Games never commented as to why the indefinite closure of such a program, but we can only speculate and assume that



either it was too difficult to automate or it simply wasn't effective in achieving what they wanted. Since the closure of Tribunal, the only way to retrieve any chatlogs is through Riots Customer Support, where the player has to pay a small fee in order to gain access. Even then, players are only given access to chatlogs for the games they participated in, which would limit the type of chatlogs and variety that one could train and test with.

There was also no way for us to analyze whether the toxicity presented in the chatlogs was provoked or a response to another action that occurred, whether that be in chat or in the actual gameplay. The point of view of the toxic messages is important, as it could change the overall definition of toxicity.

The definition of toxicity was also something that may have limited the findings. While lot of terminology has changed, and just like we hypothesized, gaming during and after the COVID-19 pandemic has truly changed the definition of toxicity and made it more intense. The findings from our sentiment analysis showed that almost everyone was on a similar toxicity level, which was very different than what was hypothesized. It could, in fact, still stand true that even in today's games everyone is at a similar toxicity level, but data would need to be properly collected in order to confirm or deny that.

## VI. IMPLICATIONS

From the evaluation that was conducted, there does not seem to be a link between toxicity and reporting a player. From our evaluation, we can see that there doesn't seem to be a link between toxicity and reporting a player. This shows that maybe the game as a whole as reached a level of toxicity where simply trying to look at the most "toxic" person does not result in players wanting to report you. Though this could also mean that there are more gameplay based factors that resulted in the player getting reported instead of solely being based on the chat messages. As mentioned in limitations, there are many false positives that could be occurring.

Game companies could use this data and work to further their progress with finding methods to help make the game community less toxic and instead more friendly, which would help newer players come in. The type of work conducted here would benefit everyone, both users and even developers, as more players would be in check of their attitudes in chat and in game, which could result in less reporting and banning instances where the developers have to jump in to handle.

No one would be affected in a negative way, the only players that this software would really affect are those that approach the threshold of toxicity, and even then the benefits of being warned would be to avoid reports and eventual bans against the user's account.

## VII. CONCLUSION

Creating software to manage and handle player reports and toxicity is a must for game developers as the E-sports industry continues to grow exponentially. Datasets need to be made accessible so that outside research can be conducted to help

further the findings and to attempt to find a solution. By leveraging data collected through League of Legends Tribunal services, along with a variety of natural language processing techniques, this software is able to attempt to find correlation between messages sent in chat during a game and the resulting reported player.

Though no correlation was found, this isn't the end of research in this field. Future work lies in the hands of gaming companies releasing new data on their chat and report systems. Since the release of the dataset, E-sports have grown to be one of the top industries, with world competitions gathering masses of people larger than any other type of event in history.

There are multiple different ways that this same problem could be approached that could potentially lead to better results and thus better correlations. Though these methods heavily rely on having more data.

Future work could approach with a different scope and try to isolate specific games to really grasp the patterns on how toxicity may result in a report. These games would need to be verified games that League themselves agree with the reporting. Training on verified data could result with better pattern recognition, thus models having a better grasp as for what to look at. We fear that due to the nature of the dataset and false positives, it is possible we trained with too many false positives, thus creating errors when testing.

Finding a way to analyze and include gameplay details would also help provide more features for the models to be able to work off of. Features such as initial assigned role and position in the game versus what character the player actually chose to play could add to why players would be willing to report someone. If the character and the role they play don't line up, that could indicate further gameplay that could affect the reports for the game.

Having more knowledge on the stats of the game could help limit false positives as well as provide more features. With stats such as the damage done in game could shine light on any outliers in the game, such as new players that falsely get reported or even those players that leave in the middle of a game.

Furthering development of the models and sentiment analysis tools that were used could also improve results; especially as technology grows and tools such as sentiment analysis and auto-correct are able to more accurately read data and specify their output according to the type of data. Companies will need to explore solutions to toxic game environments if they want to continue to create a space where players can not only have fun, but also thrive and grow.

## REFERENCES

- [1] Statt, N. (2021, December 31). New Unity Study shows just how toxic online gaming can be. Protocol. Retrieved April 2022, from <https://www.protocol.com/unity-online-gaming-toxicity-study>
- [2] The Harris Poll. (2021). Toxicity in multiplayer games report. Unity. Retrieved April 2022, from <https://create.unity.com/toxicity-in-multiplayer-games-report>

- [3] Paraggua, B., Nieva, J., amp; Sharma, A. (2021, December 29). New World: Mass reporting is a huge problem that developers must fix. Player.One. Retrieved April 2022, from <https://www.player.one/new-world-mass-reporting-huge-problem-developers-must-fix-144602>
- [4] Cote, A. (2021, October 14). A toxic culture: How covid-19 has caused youth harassment to skyrocket in online gaming. The Milwaukee Independent. Retrieved April 2022, from <https://www.milwaukeeindependent.com/syndicated/toxic-game-culture-covid-19-caused-youth-harassment-skyrocket-online-gaming/>
- [5] Nelson, C. (2018, October 1). To combat toxicity, game developers are censoring users - should they? Study Breaks. Retrieved April 25, 2022, from <https://studybreaks.com/tvfilm/to-combat-toxicity-game-developers-are-censoring-users-should-they/>
- [6] Anti-Defamation League. (2019, July). Free to play? hate, harassment, and positive social experiences in online games. Anti-Defamation League. Retrieved April 2022, from <https://www.adl.org/free-to-play>
- [7] League of Legends Tribunal Chatlogs. (2020). <https://www.kaggle.com/datasets/simshengxue/league-of-legends-tribunal-chatlogs>