DETECTING TOXIC COMMENTS USING MACHINE LEARNING

1st Tien V. Nguyen
University of Information Technology
Vietnam National University, Ho Chi Minh City, Vietnam
20520805@gm.uit.edu.vn

2nd Khanh P. Doan
University of Information Technology
Vietnam National University, Ho Chi Minh City, Vietnam
20521443@gm.uit.edu.vn

***Abstract*—In recent times, the rapid growth and ubiquitous presence of the internet have significantly propelled the prosperity of the gaming industry on a global scale. This unprecedented growth has attracted an extensive demographic of young individuals who actively participate in gaming activities as players and viewers. However, within this enthusiastic community, a notable segment exhibits an inability to exercise control over their linguistic expressions, leading to the proliferation of a substantial volume of toxic and harmful comments. The present research endeavor centers around the construction of a comprehensive and self-curated dataset explicitly tailored to addressing the challenge of toxic speech detection in the gaming domain. This dataset encompasses an extensive collection of over 3130 user comments, diligently gathered and meticulously annotated to label instances of toxic speech accurately. Subsequently, two prominent machine learning models, namely logistic regression and support vector machines (SVM), are trained on this expertly curated dataset.**

I. INTRODUCTION

Nowaday, online games are very popular with a variety of players of all ages. In the playing matches of the game, some players are too eager to win or are dissatisfied with certain actions of other players, using obscenity and insults to the other players, causing unnecessary conflicts among players, which makes the gaming experience worse. Moreover, this can result in worse thing : the conflict in game turn into the conflict in the real life. Then, they may solve it by hurting or even killing other. This occured many time in the past. Therefore, it is necessary to have a tool to detect obscene and offensive comments so that they can be prevented in a timely manner.

Toxic speech detection is a fundamental task in natural language processing (NLP) that plays a vital role in maintain the quality of online discussions. Besides, toxic comments, which cause the shutting down of user comments completely, are rising dramatically. Hence, filtering toxic comments relied on its level helps improve online conversations. In this paper, we introduce our dataset for identity toxicity of Vietnamese comments on social media, extremely in game domain. To ensure the quality of the dataset, we build a detailed and clear annotation scheme. We also use machine learning methods to build model for evaluating this dataset. All model work effectively on this dataset and they can be applied in the practical.

Describe the problem:

- Input: a Vietnamese comment.
- Output: assign as 1 if the comment is offensive, otherwise assign 0.

Figure 1 shows the input and output of the problem.

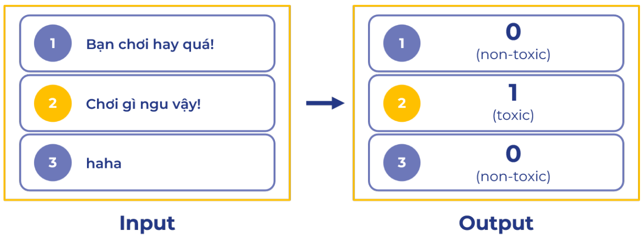

Figure 1: **The illustration of the input and output of the problem**

II. RELATED WORK

Toxic speech detection is a critical task in the context of online communication platforms, social media, and digital communities. It involves the development of algorithms and models to identify and classify instances of toxic or harmful language automatically. With the increasing prevalence of online interactions, there is a growing need to mitigate the negative impact of toxic speech on individuals and communities.

Research on toxic speech detection has gained significant attention in recent years, aiming to create safer online environments. The detection process involves analyzing text data and applying machine learning techniques to classify content as toxic or non-toxic. Various approaches have been explored, including rule-based methods, supervised learning algorithms, and more advanced techniques such as deep learning.

- Luan Thanh Nguyen used PhoBERT on UIT-ViCTSD dataset for constructive and toxic speech detection [1].
- Hung P. Nguyen compared using machine learning and deep learning on detecting hate-speech [2].
- Fahim Mohammad compared traditional machine learning methods, from Naive Bayes (NB) and logistic regression (LR) to state of art neural networks. The performances of BiLSTM and NB-SVM are good and robust after data preprocessing compared to other models [3].

- Zimmerman et al investigated ensemble models with different hyperparameters [4].

These inspired us to combine various model architectures and different word embeddings for toxic comment classification.

Detecting toxic speech in different languages presents unique challenges. Language-specific characteristics, cultural nuances, and the availability of labeled datasets impact the performance of detection models. In the case of the Vietnamese language, specific linguistic aspects, such as diacritics and tonal variations, add complexity to the detection process.

III. DATASET

A. Data Collecting and Preprocessing

We build the dataset for this project by ourselves from initial. The dataset just pays attention to Vietnamese language. First at all, each data is collected from various sources such as: crawling data in directly games, in posts on social media, in the comments of live streams. The main source is in the directly games, which brings more natural context. The first 10 rows of the dataset are shown in Figure I.

Table I: 10 Head rows of dataset.

text	label
xin mid gánh team với	0
bạn ơi	0
cho mid là win	0
câm à	1
cầm thanh tẩy	0
15 ff	0
vcl sợ thế	1
vãi cả lồn sợ thế	1
bên nào ff	0
nói cho rõ vào	0

The dataset we built is a collection of texts, and it is an unstructured data, which is why it is difficult to be approach with models without applying preprocessing methods. So in the next step, we preprocessed texts in corpus. It includes:

- Converting all letters of comments to lowercase situation.
- Removing teencode and stopwords.
- Replacing acronyms with complete words and correct spelling mistakes.
- Removing extra spaces.
- Deleting intentionally duplicated words.
- Adding cleaned comments to the data.

The above actions can improve computational time. The more cleaning data, the less dimensions and noise.

B. Data Labeling

The dataset has 3130 rows of data, which has two column : one is text and another is label. Text column contain comments that is collected from game matches. Label column is the label of data that we will predict. It includes two value : 0 and 1. We will assign a comment as 0 or 1 based on if there are obscene or insults words. If the text has, it will be assign to 1, otherwise it will be 0.

The details of our labels are as follows: The condition for us to label toxic is that in the comment line there are swear words, or words that offend others. Comments that do not fall into the above two cases will be labeled non-toxic. There are a few sentences that are critical or reproachful but do not contain the words in the above 2 cases, they are still considered non-toxic labels.

The codebook of dataset is in Table II.

Table II: Codebook of dataset.

No	Field	Description	Value Range
1	text	comment of player	all Vietnamese string type
2	label	toxic or non-toxic	0 or 1

After completing building the dataset, we split it into three sets: a training, a validating and a test set with a ratio of 7:2:1.

C. Dataset Evaluating

Every row is assigned label by two annotators. Each annotator works independently to ensure the objectivity of labeling. After finishing labeling process, we use Cohen's Kappa [5] to measure the agreement of the dataset. Calculation of Cohen's Kappa is performed according to the following formula:

$$k = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)} \tag{1}$$

where

Pr(a): the actual observed agreement

Pr(e): chance agreement

Cohen suggested the Kappa result be interpreted in Table III.

Table III: Kappa Result. [6]

Value of Kappa (k)	Level of Agreement
0 - 0.2	Poor
0.21 – 0.39	Minimal
0.4 - 0.59	Weak
0.6 – 0.79	Moderate
0.8 – 0.9	Strong
Above 0.9	Almost Perfect

After calculating, the result is 0.95. It is high score, so we can believe that the dataset is trustworthy. In the end, we found that our dataset has 1183 labels 1 and 1947 labels 0. This dataset is balance, which may give a good result when applying classification. Figure 2 shows the distribution of data labels.

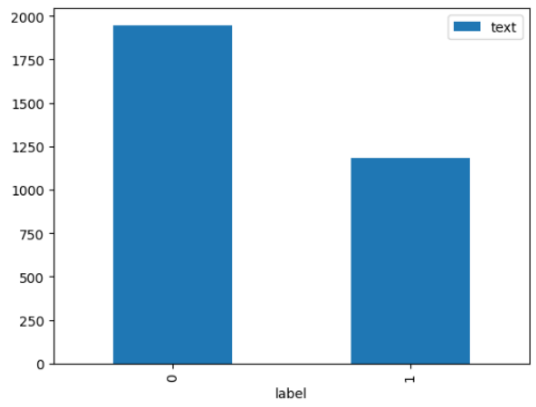

Figure 2: **The distribution of label empty citation**

IV. METHOD

A. Work Flow

There are three phases we went through to solve this problem. Figure 3 shows an overview of our proposed system described as follows.

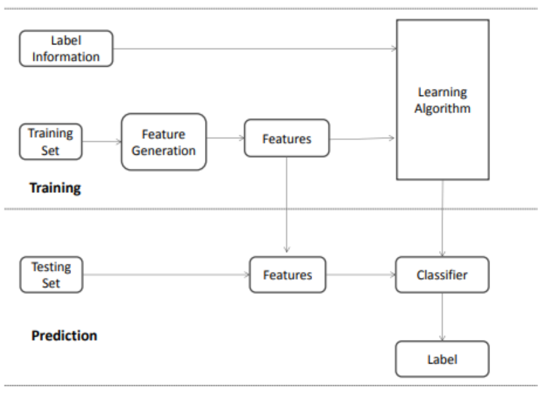

Figure 3: **The step by step implementation**

First of all, we preprocessed texts in corpus. It includes converting all letters of comments to lowercase situation, removing teencodes and stopwords to reduce data dimentions [7], which can improve computational time. The more cleaning data, the less dimensions and noise.

Secondly, we create feature vector. From the cleaned dataset, we built a dictionary of words appeared in sentence. Then, we change text data to numeric data by creating feature vector based on the dictionary . We used tf-idf, which is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus [8], to extract number to feature vector. If the dimensions in above feature vector we created is too large, the computational time will be longer.

Finally, we applied machine learning model to predict label of data. Every time we pass to new comment, we also change it to feature vector. After the models were trained, we evaluated it on the validating set. Then, to improved performance of models, we optimize hyperparameters of models by using GridSearchCV to choose the best parameters of each model. We evaluated it again on validating set. In the end, we tried models on test data.

B. Feature Extraction

1) TF-IDF + DNN: Our method use TF-IDF as a word embedding module

- **TF-IDF**: Stands for Term Frequency Inverse Document Frequency of records. It can be defined as the calculation of how relevant a word in a series or corpus is to a text. The meaning increases proportionally to the number of times in the text a word appears but is compensated by the word frequency in the corpus (dataset).
  - **Term Frequency**: In document d, the frequency represents the number of instances of a given word t. Therefore, we can see that it becomes more relevant when a word appears in the text, which is rational. Since the ordering of terms is not significant, we can use a vector to describe the text in the bag of term models. For each specific term in the paper, there is an entry with the value being the term frequency. The weight of a term that occurs in a document is simply proportional to the term frequency.

$$tf(t,d) = \frac{\text{count of } t \text{ in } d}{\text{number of words in } d} \tag{2}$$

  - **Document Frequency**: This tests the meaning of the text, which is very similar to TF, in the whole corpus collection. The only difference is that in document d, TF is the frequency counter for a term t, while df is the number of occurrences in the document set N of the term t. In other words, the number of papers in which the word is present is DF.

$$df(t) = \text{ occurrence of } t \text{ in documents} \tag{3}$$

  - **Inverse Document Frequency**: Mainly, it tests how relevant the word is. The key aim of the search is to locate the appropriate records that fit the demand. Since tf considers all terms equally significant, it is therefore not only possible to use the term frequencies to measure the weight of the term in the paper. First, find the document frequency of a term t by counting the number of documents containing the term:

$$df(t) = N(t) \tag{4}$$

where
df(t) = Document frequency of a term t
N(t) = Number of documents containing the term t

Term frequency is the number of instances of a term in a single document only; although the frequency of the document is the number of separate documents in which the term appears, it depends on the entire corpus. Now let's look at the definition of the frequency of the inverse paper. The IDF of the word is the number of documents in the corpus separated by the frequency of the text.

$$idf(t) = \frac{N}{df(t)} = \frac{N}{N(t)} = log(\frac{N}{df(t)}) \quad (5)$$

– **Computation**:

$$tf - idf(t,d) = tf(t,d) * idf(t) \quad (6)$$

C. *Models*

We used two methods : logistic regression and support vector machine (SVM) (Crammer and Singer, 2001) to build model for detecting toxic comment. These are popular and simple models that is suitable for classification. For the implementation, we used sklearn library, which is simple and strong library for machine learning.

1) SVM: Support Vector Machines (SVM) is a powerful classification model widely used in machine learning and pattern recognition tasks. It is particularly effective for solving binary classification problems, where the goal is to separate data points into two distinct classes.
The fundamental principle behind SVM is to find an optimal hyperplane that maximally separates the two classes while minimizing the classification error. This hyperplane acts as a decision boundary, with data points falling on one side belonging to one class and those on the other side belonging to the second class.
Mathematically, SVM aims to find the hyperplane defined by the equation:

$$w^T x + b = 0 \quad (7)$$

Where 'w' is a weight vector perpendicular to the hyperplane, 'x' represents the input feature vector, and 'b' is the bias term. The weight vector 'w' and bias term 'b' are learned during the training process to determine the optimal hyperplane.
The classification decision is made based on the sign of the equation:

$$f(x) = sign(w^T x + b) \quad (8)$$

If f(x) is positive, the data point 'x' is classified as one class; if it is negative, it belongs to the other class. The f(x) magnitude represents the data point's distance from the decision boundary.
An optimization algorithm is employed to find the optimal values of 'w' and 'b' to train an SVM model. The objective is to maximize the margin, which is the distance between the hyperplane and the closest data points from each class. This optimization process involves minimizing a cost function that penalizes misclassified data points and maximizes the margin.

2) Logistic Regression: Logistic Regression is a popular classification model used to predict the probability of an event occurring. It is widely employed in machine learning and statistics for binary classification problems, where the output variable has two distinct classes.
The core idea behind logistic regression is to model the relationship between the input features and the probability of belonging to a particular class. It uses the logistic function, also known as the sigmoid function, to transform the linear combination of input features into a probability value between 0 and 1.
The Logistic Function (Sigmoid Function) is a simple S-shaped curve used to convert data into a value between 0 and 1. The Figure 4 is showing the logistic function:

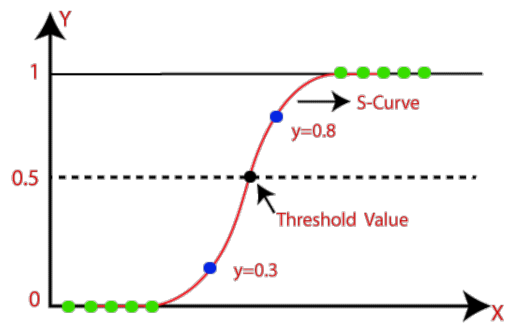

Figure 4: **The Logistic Function [9]**

Mathematically, the logistic regression equation is defined as:

$$p(y = 1 \mid x) = 1/\left(1 + e^{-x}\right) \quad (9)$$

where p(y = 1 | x) represents the probability of the output variable 'y' being 1 given the input features 'x'. The term 'z' is the linear combination of input features weighted by corresponding coefficients:

$$z = w^T x + b \quad (10)$$

Here, 'w' represents the weight vector, 'x' is the input feature vector, and 'b' is the bias term.
During training, the logistic regression model learns the optimal values for the weight vector 'w' and bias term 'b'. This is achieved by maximizing the observed data likelihood given the model's parameters. The optimization algorithm, such as gradient descent, minimizes the cost function and updates the parameter values iteratively.
To make predictions, the logistic regression model uses a decision threshold. If the predicted probability is above the threshold, the instance is classified as class 1; otherwise, it is classified as class 0.

V. EXPERIMENTS AND RESULT

We splited dataset to training, validating and testing. Then, we trained models, evaluated and optimized on validation set. In the end, we tried both model on test data. There are some metrics for evaluation, as showing in the following, but in this study, we just use F1-score to evaluate the performance of models.

A. Metrics

1) Confusion Matrix:: A confusion matrix is a tabular representation of the model's predictions compared to the ground truth labels. It consists of four elements:

- True Positives (TP): Instances correctly classified as positive.
- False Positives (FP): Instances incorrectly classified as positive.
- False Negatives (FN): Instances incorrectly classified as negative.
- True Negatives (TN): Instances correctly classified as negative.

2) Precision:: Precision measures the proportion of true positive predictions among all positive predictions made by the model. It is calculated using the following formula: Precision = TP / (TP + FP)

3) Recall:: Recall quantifies the proportion of true positive predictions among all actual positive instances in the dataset. It is calculated using the following formula: Recall = TP / (TP + FN)

4) F1 Score:: The F1 score combines precision and recall into a single value, providing a balanced measure of a model's performance. It is calculated using the following formula:
F1 Score = 2 * (Precision * Recall) / (Precision + Recall)
The F1 score is the harmonic mean of precision and recall, emphasizing the importance of both metrics. It ranges from 0 to 1, where 1 represents perfect precision and recall.

B. Result

We use metrics as macro-averaged F1-score for evaluating the performances of models. The result running on test and validation is presented in the Table IV.

Table IV: The result running on validation and test.

Models	Valid	Test
Logistic Regression	0.882	0.769
SVM	0.947	0.769

With the above results, we get quite satisfactory F1-score results on 2 datasets: valid and test. In the Logistic Regression model, the result is 88% on the val set and about 77% on the test set; In the SVM model, the result on the val set is about 95%, higher than the LR model and about 77% on the test set.
Although they perform well , we want to increase model's accuracy. One of the methods is optimize hyperparameter [10]. Sklearn also support to do this by using GridSearchCV. It can find a set of best parameter in function to improve model depending on the data. The GridSearchCV function will concatenate each parameter's value together and choose a set of value which give the best performance. The result is in the Table V.

Table V: The result running on test and validation using GridSearchCV.

Models	Valid	Test
Logistic Regression	0.952	0.814
SVM	0.952	0.814

After using GridSearchCV, we get better results. And the results of both models seem to be comparable.

C. Evaluation

As we can see in Table III, IV, the models may overfitting. The score is high on validation data but it descreased when running on test data. One of the reason may the dataset too small and lacking of data. The solution is adding more data to corpus to improve the accuracy on test set.

VI. CONCLUSION AND DEVELOPMENT

Detecting toxic comments in time and blocking them is important. It may increase user's experience and build the healthy culture of behavior in game. In this study, we introduced how to detect toxic comment using machine learning. We have built a dataset including many comment by Vietnamese language from many sources and then we applied logistic regression and support vector machine to build classification model that evaluates effectively on this dataset. To improve precision of the model, we optimized hyperparameter of each model. The performance increased a little.
In the future, we will try to make model better by adding more data to the dataset, expand it not only in game but also in other subject and apply deep learning for having more experiment to get the best result.

VII. ACKNOWLEDGMENT

We would like to express our sincere thanks to each team member who contributed to the completion of this study. Our team conducted a clear assignment of work, collected data sets, agreed on the evaluation protocol,

and tested experimental results. Details of the work are shown in the Table VI.

Table VI: Tasks assignment.

Tasks	**Tien V. Nguyen**	**Khanh P. Doan**
Building dataset	x	x
Coding	x	
Report	x	x
Slide		x
Presenting	x	x

References

[1] N. L.-T. N. Luan Thanh Nguyen Kiet Van Nguyen, “Constructive and toxic speech detection for open-domain social media comments in vietnamese,” 2021.

[2] N. L.-T. N. Hung P. Nguyen Kiet Van Nguyen, “Comparison between traditional machine learning models and neural network models for vietnamese hate speech detection,” 2020.

[3] F. Mohammad, “Is preprocessing of text really worth your time for toxic comment classification? int’l conf. artificial intelligence,” 2018.

[4] U. K. Steven Zimmerman Chris Fox, “Improving hate speech detection with deep learning ensembles,” 2018.

[5] M. L. McHugh, “Interrater reliability: The kappa statistic,” 2012.

[6] M. L. McHugh, “Interrater reliability: The kappa statistic,” 2012.

[7] A. I. Kadhim, “An evaluation of preprocessing techniques for text classification,” *International Journal of Computer Science and Information Security*, 2018.

[8] S. C. Bijoyan Das, “An improved text sentiment classification model using tf-idf and next word negation,” 2018.

[9] *Logistic regression in machine learning*. [Online]. Available: https://www.javatpoint.com/logistic-regression-in-machine-learning.

[10] A. S. Li Yang, “On hyperparameter optimization of machine learning algorithms: Theory and practice,” 2020.