Sentiment Analysis of Twitch.tv Livestream Messages using Machine Learning Methods

Aryan Chouhan
Computer Engineering Department
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
0000-0001-8004-7586

Aayush Halgekar
Computer Engineering Department
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
0000-0002-3515-5665

Ashish Rao
Computer Engineering Department
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
0000-0002-6676-8819

Dhruvi Khankhoje
Computer Engineering Department
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
0000-0003-3238-7687

Meera Narvekar
Computer Engineering Department
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
0000-0003-4602-4094

2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT) | 978-1-6654-1480-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICECCT52121.2021.9616932

***Abstract*—Livestreaming refers to the practice of broadcasting media over the internet in real-time. It has steadily grown in popularity with platforms such as Twitch.tv seeing a consistent influx of users over the past few years. Twitch provides a chat feature that allows viewers to participate by sharing their opinions in real-time. These messages can serve as rich and interesting banks of public thought that can help streamers make decisions synchronously about the content that they are producing. However, with several thousand viewers involved, the added complexity of context-specific emotes, memes and the meta-language that pervades through these platforms, summarizing and extracting useful information from them becomes a daunting task. Sentiment analysis is a possible solution to this problem. In this paper, a methodology to perform sentiment analysis with a set of machine learning-based models on livestream messages from Twitch.tv is proposed. Machine Learning models like Support Vector Classifier, Logistic Regression, Decision Tree Classifier, Random Forest Classifier and Multinomial Naïve Bayes were implemented. Among these models, the Support Vector Classifier outperformed the current state-of-the-art model and displayed a 10.3% rise in accuracy, a 9.1% rise in recall and a 7.5% rise in the F1-score.**

Keywords—Machine Learning, Natural Language Processing, Sentiment Analysis, Livestreaming, Supervised Learning

I. Introduction

The online gaming industry has exploded in popularity over the past few years. With this rise, a parallel vertical has emerged – that of watching other people play video games, now referred to as 'livestreaming'. Schematically, livestreaming involves the broadcasting of a live video feed from the device of the host (called a 'streamer') by sending it to a public platform that consequently relays it to viewers across the world.

This work centers around Twitch.tv, one of the largest livestreaming platforms available today. Twitch offers a service that allows anybody to set up their own livestreams from anywhere. Although video games are the main commodity consumed, content on Twitch has diversified to other creative content such as playing instruments, making art and other forms of lifestyle content too, such as vlogging (video blogging).

Twitch also features a chat section along with every stream that allows users to communicate and interact with other viewers in real-time. Since this feedback is live and instantaneous, it contains relevant information about the stream itself. For instance, a period of high excitement in a game will usually correspond to an increase in the number of messages sent and in the consistency of the sentiment expressed via the messages.

This form of information exchange is not without its faults. With an increase in the number of viewers, it becomes increasingly difficult for the streamer to dissect useful and genuine messages. Moreover, the language on Twitch has evolved to develop a culture of its own involving convoluted slang and specific emoticons. These can vary from situation to situation (for example, an embarrassing situation might call for a specific expression, while a moment of luck might call for another) and from streamer to streamer, that is, the same sentiment when expressed on a different stream might evoke a different response. This leads to hyper-specific contextualization that makes it difficult to discern the context using only the message.

Twitch is also famed for its widespread use of 'emotes'. These are Twitch-specific emoticons and are of two classes – global and custom – which further increase the variance in expression between different channels. The language also consists of many abbreviations, repetitions and deliberate grammatical mistakes. All of these factors in conjunction make it increasingly difficult to analyze and process information from these chats.

To analyze this information, the proposed approach utilizes sentiment analysis. By grouping messages into different classes as per their subjective state, tangible information can be provided to streamers regarding the public opinion surrounding the events occurring in the stream. This information can be used to inform further choices regarding certain actions and the emotion they evoke.

The key objectives and contributions of this work involve:

- The application of various machine learning models on complex Twitch data to analyze sentiments of the users viewing the livestream.
- Successfully training a model that outperforms existing approaches.

The paper discusses previous work surrounding the subject in Section 2. In Section 3, the proposed design is highlighted with a delineation of each method used. Section 4 summarizes the results by comparing different models while Section 5 concludes the work with an outline of future research directions.

II. Related Work

The problem of sentiment analysis has been traditionally attacked through lexicon-based and supervised learning-based

approaches. For example, in a social media context, VADER (Valence Aware Dictionary and Sentiment Reasoner) [1] provides comprehensive lexicon and rule-based sentiment analysis. Recently, however, deep learning-based approaches have been gaining momentum with the increase in computing power availability. [2] and [3] summarize the state of the field and [4] and [5] focus on surveying deep learning-based approaches in detail.

Research in this area has also attempted to tackle the added complexity resulting from emojis when analyzing text. [6] proposed an Emoji Sentiment Ranking system as a European language-independent resource for automated sentiment analysis. On the other hand, emoji2vec [7] captures word embeddings of all Unicode emojis (similar to how word2vec [8] captures word embeddings), which can consequently be used in downstream NLP tasks. Tangentially, [9] uses emoji occurrences to learn domain-independent emotional representations.

Livestreaming as a culture has also attracted considerable research interest owing to idiosyncrasies that set it apart from platforms that provide traditional social media experiences such as Twitter and Facebook. As an interactive content-delivery system revolving around immediate feedback, it has engendered unique community experiences that serve as rich social dynamics investigation areas. [10], [11] and [12] explore this phenomenon and provide comprehensive reviews of how such communities form and function, while [13] analyses audience-streamer dynamics in depth.

The language used on Twitch is also of particular interest, given its diversity, abundant neologisms and peculiar site-specific emoticons (called 'emotes'). Through qualitative surveys, [14] attempted to find emote context and study how they affect the participatory community. [15] used descriptive and corpus-based methods to examine the variety of language used by communities on Twitch.

While [16] uses a hybrid of statistical and rule-based models to analyze the sentiment of comments on Twitch, the use of machine learning methods has been quite limited. [17] defines two tasks: one, predicting emotes that are likely to be used in a comment and two, detecting troll messages, for which they utilize bidirectional Long-Short-Term-Memory neural network. [18] proposes unsupervised lexicon-based and weakly supervised neural network-based approaches for analyzing the overall sentiment of individual comments. With the help of a manually created emote lexicon and a weakly supervised learning scheme, they achieved an accuracy of 63.8% at identifying Twitch comments' sentiment.

This paper takes this work forward by applying a broader class of machine learning algorithms to solve the problem at hand. Classic classification algorithms such as Logistic Regression and Support Vector Classifiers are used, amongst others. Using these methods, sentiment analysis models are built that perform significantly better than existing architectures.

III. Proposed Design

First, the data is pre-processed, after which it is split into three different corpora for training, validation and testing. After this, feature extraction is carried out and finally, the models are trained and evaluated. Fig. 1 displays the process design.

A. Dataset Description

To train the model, a labeled dataset made available by [18] was used. The dataset contains 1922 comments, of which 770 (40.06%) were classified as positive, 404 (21.02%) as negative and 748 (38.92%) as neutral.

The comments were sampled from an unlabeled dataset of around 14.4 million comments using a weighted sampling scheme. These were from the five most commented English Twitch channels from May 2018 that were not dominated by bots. The sampled data was first cleaned and anonymized. It was then annotated via a crowdsourcing campaign. Please refer to [18] for a detailed account of the sampling mechanism.

B. Pre-processing

The preprocessing pipeline contained three steps – tokenization, removal of stop words and lemmatization. To illustrate this, consider the following sentence – 'I am loving this game soooooooo much Kappa'.

1) Tokenization: For tokenization and basic preprocessing, the same strategy was employed as [18]. The sentences were first tokenized using comprehensive regular expressions. These were made to account for emotes, emoticons, numbers, Twitter-like hashtags, HTML tags and

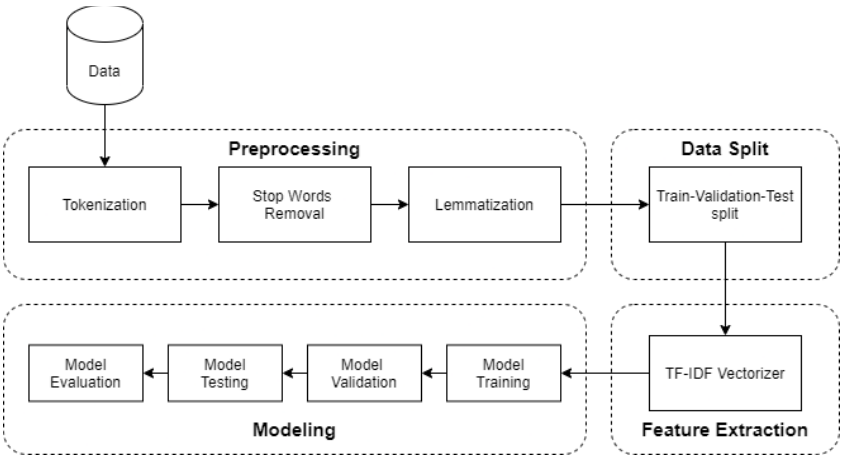

Fig. 1. Process Workflow

normal words. Then, by scanning through each word of a message, everything except emoticons was converted to lower case. The emotes were identified by cross-referencing them with a lexicon created by [18] that contained the top 100 emotes measured by their usage frequency from the unlabeled dataset. Finally, occurrences of hyperlinks were replaced with "URL" and characters occurring more than twice in succession were reduced to two occurrences. After this stage, the original sentence 'I am loving this game soooooooo much Kappa' is converted to a list with the following contents '['i', 'am', 'loving', 'this', 'game', 'soo', 'much', 'kappa'].

2) Stop Words: Stop words comprise common words that provide little to no meaning towards analyzing text. These words are dropped from the sentences to ensure that the model focuses on essential words only. NLTK's [19] default stop word list with one modification adopted from [18] was used – conjugates of stop words that contain apostrophes by removing the apostrophes were added. This is because it is common to misspell words in a live chat environment. On removing the stop words from the example above the list is now of the form '['loving', 'game', 'soo', 'much', 'kappa']'.

3) Lemmatization and/or Stemming: Lemmatization and Stemming are similar practices insomuch as they are both used to derive root words from their inflected forms. Lemmatization takes into account the morphological analysis of the words while stemming simply removes the suffix from it. Using only one of these methods is sufficient for efficient preprocessing. The WordNetLemmatizer from NLTK [19], based upon Princeton's WordNet [20], [21] was used to lemmatize the data. On using lemmatization on the example above, the final sentence takes the form '['love', 'game', 'soo', 'much', 'kappa']'.

C. Data Split

In this step, the data is split into three distinct sets. They are summarized in Table 1.

D. Feature Extraction

Given that all inputs for a model must be tensors of numeric values, the text must be vectorized. There are two well-known methods to do this – TF-IDF vectorization and Count vectorization.

1) Count Vectorization: This method involves calculating individual tokens' frequency in the entire document and using these frequencies to encode the text into integers.

2) TF-IDF (Term Frequency – Inverse Document Frequency) Vectorization: This method involves applying Count Vectorization and then normalizing the count vector using a TF-IDF transformation.

TABLE I. DATA SPLIT

Dataset	Size (% of original dataset)	Purpose
Train set	60%	Model training
Validation set	20%	Hyper-parameter tuning
Test set	20%	Model evaluation

This is carried out to re-weight the counts (preventing high-frequency words from becoming too favored and hence, preventing imbalance) by assigning them a score. TF-IDF vectorization was used in the proposed approach because it allows us to rank the importance and relevance of each word which improves performance by ignoring impertinent data.

E. Modelling

This system was tested using different machine learning classification algorithms that were used to predict the sentiment of the streamed message. The Python library, scikit-learn [22] was used for modeling purposes.

The models used are – Logistic Regression, Multinomial Naïve Bayes, Support Vector Classifier, Random Forest Classifier and Decision Tree Classifier.

IV. RESULTS AND DISCUSSIONS

In this section, different models are compared and analyzed using certain metrics.

A. Model Performance

To evaluate the models, their performance against the test set was measured, the results of which summarized in Table 2. The metrics used were accuracy, macro average precision, macro average recall and the macro average F1 score.

1) Accuracy: One of the most commonly used parameters to evaluate the performance of a model is accuracy. Accuracy is the ratio of the accurately classified samples to the total number of samples in the dataset. In the context of the dataset used for this paper, it can be expressed as the ratio of the sum of the number of true positive, true neutral and true negative classifications to the total number of samples in the dataset.

$$Accuracy = \frac{True\ Negative + True\ Positive + True\ Neutral}{Total\ number\ of\ samples}$$

2) Precision: Precision is the ratio of the count of the accurately classified samples of a class to the total number of samples of the class in the dataset. For the positive sentiment class, it can be expressed as the ratio of the count of true positive classifications to the sum of the true positive and false positive classifications.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

3) Recall: Recall is the ratio of the count of the accurately classified samples of a class to the count of the actual samples. For the positive sentiment class in the used dataset, it can be expressed as the ratio of the count of the true positive classifications to the actual count of positive samples.

$$Recall = \frac{True\ Positive}{Total\ number\ of\ actual\ Positive\ samples}$$

TABLE II. MODEL EVALUATION

Model	Accuracy	Macro Average Precision	Macro Average Recall	Macro Average F1-Score
Support Vector Classifier	*70.4%*	*74.2%*	*67.0%*	*67.3%*
Logistic Regression	69.2%	73.2%	65.5%	66%
Decision Tree Classifier	67.2%	69.1%	64.5%	64.8%
Random Forest Classifier	66.4%	71.7%	62.9%	63.2%
Sentence CNN [18]	63.8%	-	61.4%	62.6%
Multinomial Naïve Bayes	65.8%	71.2%	60.2%	61.1%

4) F1-Score: F1-Score is the ratio of twice the product of the precision and the recall of a class to the sum of the precision and recall of the class. It can also be described as the harmonic mean of the precision and the recall of a class.

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Given the imbalance in the dataset, the macro average was preferred over the weighted average since the weighted average factors in how many labels each class has and so a class with fewer labels has less of an impact on the result. By using the macro average, an attempt to smooth out some of the imbalance was made.

The macro average is defined as

$$Macro\ Average\ Precision = \frac{\sum_{i=1}^{n} Precision_i}{n}$$

where,

- n = number of classes
- $precision_i$ = precision for class i

Similarly,

$$Macro\ Average\ Recall = \frac{\sum_{i=1}^{n} Recall_i}{n}$$

The $Macro\ Average\ F1 - Score$ is the harmonic mean of the Macro Average Precision and the Macro Average Recall.

$$Macro\ Average\ F1 - Score = \frac{2 * Macro\ Average\ Precision * Macro\ Average\ Recall}{Macro\ Average\ Precision + Macro\ Average\ Recall}$$

From these results, it can be concluded that the Support Vector Classifier outperforms the rest of the models. The Support Vector Classifier model also outperforms the Sentence CNN-based [23] weakly supervised classifier built by [18] by a reasonable margin. It has a 10.3% increase in accuracy, an 9.1% increase in recall and a 7.5% increase in the F1-Score.

B. Model Performance

To analyze the model even further, a confusion matrix was used to identify where it excels and where it falters. Fig 2 illustrates confusion matrix for the Support Vector Classifer.

The model excels at identifying neutral messages (with sentiment 0) and positive messages (with sentiment 1) but falters at identifying negative messages (with sentiment -1). The model's inability to classify negative messages well is concerning, yet expected.

The dataset used, as mentioned earlier, is heavily imbalanced and contains almost half the number of negative labels as opposed to positive and neutral labels. This discrepancy flows into the mechanics of the model and causes the inconsistency. Overall, this issue can be solved by further strengthening the dataset and increasing its size.

V. CONCLUSION AND FUTURE WORK

The modeling techniques proposed in this paper improve the accuracy with which sentiment analysis is performed on messages from Twitch livestreams. Several machine learning models like Support Vector Classifier, Logistic Regression, Multinomial Naive Bayes were constructed and it was observed that Support Vector Classifier performed the best with an accuracy of 70.4%. This is a 10.3% increase from the current state-of-the-art model's accuracy, demonstrating a significant improvement. The lowest accuracy of 65.8% was seen by Multinomial Naive Bayes. These machine learning models are robust and can be deployed using a variety of methods to create services that can help streamers better understand their audiences.

The current work can be extended by incorporating richer information expression such as emotion analysis. Apart from simply classifying messages as negative, positive or neutral, emoticons analysis will allow interpretation of emotions like happy, angry and sad. The messages on Twitch are ambiguous in nature and labeling the data while maintaining it's integrity can be challenging. Work can be done in strengthening the data available for research. Textual data from other popular channels and platforms can also be incorporated to increase generality and provide a more broader accuracy. Additionally, deep learning architectures like RNN and Transformers can be employed to improve the model performance even more.

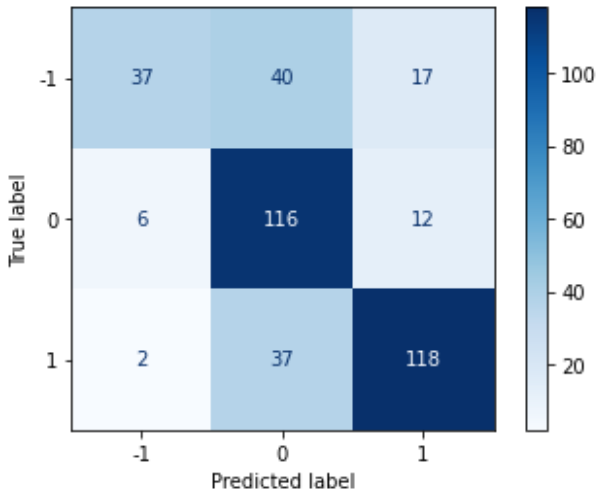

Fig. 2. Support Vector Classifier Confusion Matrix

REFERENCES

[1] C. J. Hutto and E. Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," in ICWSM 2014 - Eighth International Conference on Weblogs and Social Media, June 2014.

[2] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," Ain Shams Eng. J., vol. 5, no. 4, pp. 1093–1113, Dec. 2014, doi: 10.1016/j.asej.2014.04.011.

[3] P. Gonçalves, M. Araújo, F. Benevenuto, and M. Cha, "Comparing and combining sentiment analysis methods," in COSN 2013 - Proceedings of the 2013 Conference on Online Social Networks, 2013, pp. 27–37, doi: 10.1145/2512938.2512951.

[4] N. C. Dang, M. N. Moreno-García, and F. De la Prieta, "Sentiment Analysis Based on Deep Learning: A Comparative Study," Electronics, vol. 9, no. 3, p. 483, Mar. 2020, doi: 10.3390/electronics9030483.

[5] L. Zhang, S. Wang, and B. Liu, "Deep Learning for Sentiment Analysis : A Survey," Jan. 2018.

[6] P. Kralj Novak, J. Smailović, B. Sluban, and I. Mozetič, "Sentiment of Emojis," PLoS One, vol. 10, no. 12, p. e0144296, Dec. 2015, doi: 10.1371/journal.pone.0144296.

[7] B. Eisner, T. Rocktäschel, I. Augenstein, M. Bošnjak, and S. Riedel, "emoji2vec: Learning Emoji Representations from their Description," pp. 48–54, Sep. 2016.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in ICLR 2013 - 1st International Conference on Learning Representations, Jan. 2013.

[9] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann, "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm," EMNLP 2017 - Conf. Empir. Methods Nat. Lang. Process. Proc., pp. 1615–1625, Aug. 2017, doi: 10.18653/v1/D17-1169.

[10] M. Kaytoue, A. Silva, L. Cerf, W. Meira, and C. Raïssi, "Watch me playing, i am a professional: A first study on video game live streaming," in WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web Companion, 2012, pp. 1181–1188, doi: 10.1145/2187980.2188259.

[11] W. A. Hamilton, O. Garretson, and A. Kerne, "Streaming on twitch: Fostering participatory communities of play within live mixed media," in Conference on Human Factors in Computing Systems - Proceedings, Apr. 2014, pp. 1315–1324, doi: 10.1145/2556288.2557048.

[12] Z. Hilvert-Bruce, J. T. Neill, M. Sjöblom, and J. Hamari, "Social motivations of live-streaming viewer engagement on Twitch," Comput. Human Behav., vol. 84, pp. 58–67, Jul. 2018, doi: 10.1016/j.chb.2018.02.013.

[13] C. Flores-Saviaga, J. Hammer, J. P. Flores, J. Seering, S. Reeves, and S. Savage, "Audience and Streamer Participation at Scale on Twitch," HT 2019 - Proc. 30th ACM Conf. Hypertext Soc. Media, pp. 277–278, Nov. 2020.

[14] H. Hope, "'Hello [Streamer] PogChamp': The Language Variety on Twitch," University of Stavanger, Norway , 2019.

[15] J. Olejniczak, "A Linguistic Study of Language Variety Used on Twitch.Tv"

[16] J. M. G. B. Reis, "Sentiment Analysis: The Case of Twitch Chat - Mining user feedback from livestream chats," Mar. 2020.

[17] F. Barbieri, L. Espinosa Anke, M. Ballesteros, J. Soler, and H. Saggion, "Towards the Understanding of Gaming Audiences by Modeling Twitch Emotes," in Proceedings of the 3rd Workshop on Noisy User-generated Text, Jan. 2017, pp. 11–20, doi: 10.18653/v1/W17-4402.

[18] K. Kobs et al., "Emote-Controlled," ACM Trans. Soc. Comput., vol. 3, no. 2, pp. 1–34, May 2020, doi: 10.1145/3365523.

[19] S. Bird, E. Loper, and E. Klein, Natural Language Processing with Python. O'Reilly Media Inc., 2009.

[20] G. A. Miller, "WordNet: A Lexical Database for English," Commun. ACM, vol. 38, no. 11, pp. 39–41, Nov. 1995, doi: 10.1145/219717.219748.

[21] C. Fellbaum, WordNet: An Electronic Lexical Database, vol. 76, no. 3. Cambridge, MA: MIT Press, 1998.

[22] F. Pedregosa, "Scikit-learn: Machine Learning in Python," J. Mach. Learn. Res., vol. 12, no. 85, pp. 2825–2830, 2011.

[23] Y. Kim, "Convolutional Neural Networks for Sentence Classification," EMNLP 2014 - 2014 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf., pp. 1746–1751, Aug. 2014.