Classifying YouTube Comments Based on Sentiment and Type of Sentence

Rhitabrat Pokharel
pokharel@pdx.edu
Portland State University
Department of Computer Science
Portland, OR, USA

Dixit Bhatta
dixit@udel.edu
Department of Computer and Information Sciences
University of Delaware
Newark, DE, USA

ABSTRACT

As a YouTube channel grows, each video can potentially collect enormous amounts of comments that provide direct feedback from the viewers. These comments are a major means of understanding viewer expectations and improving channel engagement. However, the comments only represent a general collection of user opinions about the channel and the content. Many comments are poorly constructed, trivial, and have improper spellings and grammatical errors. As a result, it is a tedious job to identify the comments that best interest the content creators. In this paper, we extract and classify the raw comments into different categories based on both sentiment and sentence types that will help YouTubers find relevant comments for growing their viewership. Existing studies have focused either on sentiment analysis (positive and negative) or classification of sub-types within the same sentence types (e.g., types of questions) on a text corpus. These have limited application on non-traditional text corpus like YouTube comments. We address this challenge of text extraction and classification from YouTube comments using well-known statistical measures and machine learning models. We evaluate each combination of statistical measure and the machine learning model using cross validation and F_1 scores. The results show that our approach that incorporates conventional methods performs well on the classification task, validating its potential in assisting content creators increase viewer engagement on their channel.

CCS CONCEPTS

• **Information systems → Specialized information retrieval**;
• **Computing methodologies → Information extraction**.

KEYWORDS

multiclass classification, nlp, sentence classification

Reference Format:
Rhitabrat Pokharel and Dixit Bhatta. 2021. Classifying YouTube Comments Based on Sentiment and Type of Sentence. *arXiv publication.* Portland, OR, USA, 8 pages.

arXiv publication, October 2021, Portland, OR, USA

1 INTRODUCTION

In recent years, YouTube has gained huge popularity among content creators. A large number of content creators upload their video content on this platform. These videos get tons of views and comments. The content creators, more generally called YouTubers, need to continuously work on maintaining the quality and quantity of their contents. To do so, they must collect feedback from their viewers through the comments section. This feedback lets them understand the influence of their creations. In addition to improving audience engagement, feedback also provides information on the aspects of the content that need improvement.

However, not all YouTubers have enough time to go through all the comments on individual video. On the contrary, they must read all the comments to fully understand the public interest on their content. The solution to this inconvenience is addressed in our work. Our approach is to extract all the comments from a video and categorize them into multiple categories based on both sentiment and sentence types: Negative, Positive, Interrogative, Imperative, Corrective, and Miscellaneous. These categories can help YouTubers focus only on those comments that suit their interest.

There have been multiple studies in the field of sentiment analysis such as Twitter sentiment analysis [1], YouTube polarity trend analysis [12], user comment sentiment analysis on YouTube [4], and so on. However, not enough research has been carried out on sentiment analysis through classification of a sentence based on its type. We have approached this issue from the perspective of YouTube comments. Consequently, it is a challenging task to categorize the comments into different sentence types because of various factors such as non-standard language, spelling errors, unformatted texts, and trivial comments. Apart from these, sometimes there are multiple sentences of different classes on a single comment. The combination of these issues poses a unique challenge in sentiment analysis based on sentence types.

One of the simplest ways to address the problem is to categorize the comments purely based on lexicon [3] e.g., the interrogative comments can be identified from keywords such as what, how, and why. Similarly, positive sentences can be identified from keywords like good, best, and wonderful. However, this approach is naive and does not address unique challenges presented by informal texts. Moreover, this method performs poorly if a single comment comprises of multiple categories. Such comments can be be categorized more efficiently by appropriately extracting features from the text corpus and using supervised machine learning techniques [9]. Neural networks [21] can be used as a potential solution, however, they are difficult to tune and are not readily explainable. Explainability

is especially important for comments that fall under multiple categories to clearly understand why a resulting category was selected.

Our approach is to extract features from preprocessed data and use those features to train well-known supervised learning algorithms. The supervised learning models can then be fine tuned to get the best results. Since the performance of the model is dependent on the text corpus, we select multiple popular fine tuned algorithms for this task and observe their performance. We experiment our YouTube comments dataset with five different fined tuned classification models using two different feature extraction methods to obtain the best results for each model. The accuracy of the models are calculated in terms of cross validation score and F_1 score. Although our approach is simple, the results are effective enabling content creators to view their feedback of interest easily.

1.1 Organization

The rest of the paper is organized as follows: Section 2 reviews the related work on feature extraction from text, YouTube comment sentiment analysis, and sentence type categorization. Section 3 discusses our approach. Similarly, Section 4 presents the results of the approach and Section 5 concludes the paper with some remarks and future enhancements.

2 RELATED WORK

Various studies have been done on feature extraction from text. In their thesis work, Poche *et al.* [15] have studied the commenting behaviour of viewers on coding tutorial videos. Using SVM and Naive Bayes models, the comments collected were classified into two broad categories: Content Concerns and Miscellaneous. However, classifying comments into multiple sentence types is a more complex task; a detailed classification of "Content Concerns" was performed in our work with increased accuracy.

Taboada *et al.* [18] implemented lexicon based methods for sentiment analysis. In this study, a dictionary of words was created and the words were manually ranked into a scale of -5 to +5 (negative to positive sentiment). Our supervised learning approach replaces the manual work performed during the classification step of their approach, resulting in a greater accuracy. The time consuming process of manually scaling the words was also automated by our machine learning approach. Our research classifies texts based on the type of sentence as well as the sentiment. Siersdorfer *et al.* [16] used linear support vector machine along with a thesaurus to obtain the degree of negativity and positivity of each word from comments. The accuracy of their approach stands at 0.72. Even without using any thesauruses, we were able to increase the accuracy of our model for the sentiment analysis.

Likewise, Krishna *et al.* [12] used Naïve Bayes classification for polarity (sentiment) analysis on IMDB dataset. Their approach makes use of the category of each comment during the feature selection process, which means it cannot calculate the polarity of a new comment because its features cannot be calculated without knowing its category. However, our approach does not use category during feature selection process. In a slightly different approach, the sentiment on comments is used to aid in finding the relevant video on YouTube based on the search text by Bhuiyan *et al.* [4]. This approach was only able to classify the comments into positive and negative categories.

Maas *et al.* [13] introduced an advanced concept by showing the relationship among the words in their logical meaning as well as sentimental sense. The semantic aspect was trained by an unsupervised model, whereas the sentiment aspect was trained by a supervised model (SVM). Although their model gave the polarity of the text, they did not experiment with the type of sentence. Using the conventional NLP tools, we were able to classify text based on both sentiment and type of sentence.

When it comes to classifying by sentence types, Madden *et al.* [14] provided a scheme for classification of YouTube comments. They proposed that people's commenting habits differ between groups; some comment for promotion, some for providing information, and others for pleasure. Madden's work has provided 10 main categories and 58 sub categories for such comments. However, they have merely provided a schema for the classification task. We attempted to use some of their categories into the classification task. Similarly, Cheung *et al.* [5] classified questions for the data obtained from Webclopedia. Their work was limited to the classification of different types of question.

Kim [11] implemented convolution neural networks for analysing sentiment on 7 different datasets. The paper showed that hypertuned Convolution Neural Network (CNN) can produce satisfying results. The word vectors were obtained from an unsupervised neural language model *Word2vec* [6, 8, 17]. The variants of CNN models used were CNN-rand, CNN-static, CNN-non-static and CNN-multichannel. On the other hand, Hassan *et al.* [7] proposed a recurrent neural network based model called Long Short-Term Memory (LSTM) for sentiment analysis. Both of these approaches showed promising results, but they have not explored the classification of different types of sentences (like imperative, question, corrective, and sentimental). They are likewise limited to either sentiment analysis or question classification. Our research addresses much broader sentence classification.

In most of the existing work, the sentences of the imperative class have not been researched adequately. Khoo *et al.* [10] performed experiments on different models for 14 different classes of sentences (including imperative sentence types like request, instruction and suggestions). The models used for the experiment were Naive Bayes, Decision Tree, and Support Vector Machine. Support Vector Machine overshadowed all other models and had insignificant effect from feature selection. In their work, they only chose the standard response emails because these emails have well-structured sentences and few grammatical errors. It eases the classification task. However, our work consists of a large number of unstructured sentences with huge grammatical errors. Table. 1 shows the comparative study of the previous works and our approach.

3 METHODOLOGY

In our approach, we make use of python programming language as in [2] to execute the experiment. We collect comments from YouTube videos using the YouTube Data API, which are preprocessed to filter out the noise. Then, the features are extracted from the comments and fed into the classification models. The models are finally hypertuned to produce the best results.

Table 1: Summary of the related work and our approach

Study	Dataset	Categories	Methodology
Poche *et al.* [15]	Comments on coding tutorial videos	Content Concerns, Miscellaneous	SVM, Naive Bayes
Taboada *et al.* [18]	Epinions 1, Epinions 2, Movie reviews, Camera reviews	Setiments	Lexicon
Siersdorfer *et al.* [16]	YouTube comments	Sentiments	Linear SVM + thesaurus
Krishna *et al.* [12]	IMDB dataset	Sentiments	Naïve Bayes
Bhuiyan *et al.* [4]	YouTube Comments	Sentiments	Lexicon
Maas *et al.* [13]	IMDB reviews	Sentiment + Semantic	Unsupervised + Supervised models
Madden *et al.* [14]	YouTube comments	Numerous	Manual
Cheung *et al.* [5]	Webclopedia	Question types	Naïve Bayes, Decision Tree
Kim [11]	Multiple	Sentiments	Convolutional NN
Hassan *et al.* [7]	IMDB reviews, Stanford Sentiment Treebank	Sentiments	Recurrent NN
Khoo *et al.* [10]	Email conversation	14 different sentence types	Naive Bayes, Decision Tree, SVM
Our approach	YoutTube comments	Sentence type + Sentiments	Linear SVC, Logistic Regression, Multinomial NB, Random Forest, Decision Tree

3.1 Data Collection

We build our dataset by scraping YouTube comments. We use the YouTube Data API for authenticating and accessing the comments on videos [20]. First, the API is used for authentication and then the credentials obtained are fed into the comment extractor. The comment extractor then scrapes comments from the comment section by scrolling through all the comments and loading them dynamically. Fig. 1 shows the scraping process.

The dataset consists of 10,000 comments picked from different tutorial videos [1]. We chose tutorial videos for our experiments because they contain a wide variety of comments. We then manually label the comments into 6 different classes: Positive, Negative, Interrogative, Imperative, Corrective and Miscellaneous. These classes are defined based on general needs of the YouTubers. Note that further categories can be established if or as needed. These classes belong to two broader classes: Sentiment Analysis (positive and negative) and Sentence Types (Interrogative, Imperative, Corrective and Miscellaneous). Table. 2 shows different classes and the content of that class. The classes are explained in more detail next.

Positive tells that the viewers perceived the content as worthy and that the content created a positive impact on them. *Negative* provides information on what is wrong with the content and why the viewers are not attracted to it. *Interrogative* conveys viewers' doubts and questions. It is a useful feature because the content creators can increase their influence by addressing viewers' questions and issues. *Imperative* provides viewers' expectations and requests for actions. *Corrective* tells the creator about the corrections expected to be made on the video making it a very important aspect of the feedback. *Miscellaneous* includes declarative sentences and

[1]Source Code and Dataset Publicly Available at https://github.com/Rhitabrat/Youtube-Comments-Categorization

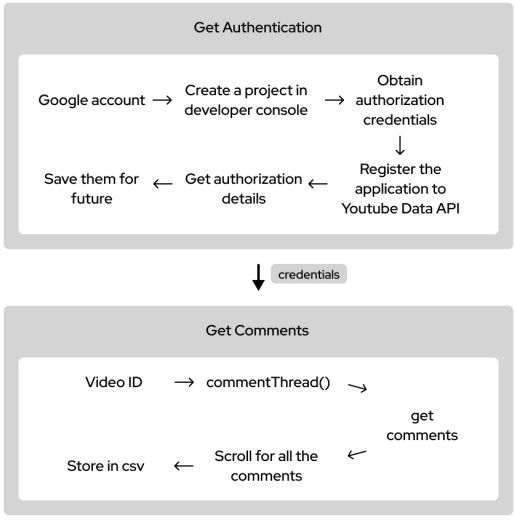

Figure 1: The process of scraping

Table 2: Classes of comments with content type

Class	Content
Positive	appraisals, appreciations
Negative	scoldings, not able to do what is told in the video
Interrogative	all type of questions, queries, asking for something, sentences starting with modal/auxiliary verbs
Imperative	requests, commands, expectations
Corrective	amendments, improvements, mistakes, remedy
Miscellaneous	remaining types, promotions, chitchat

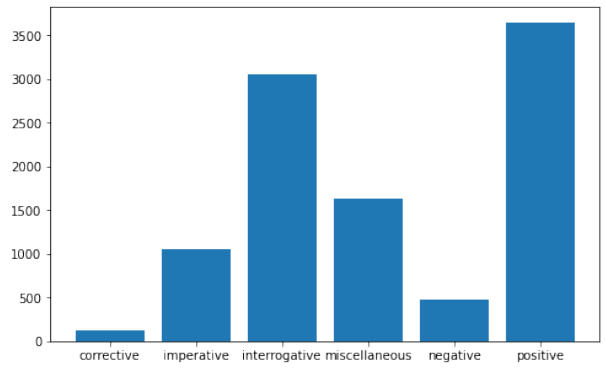

Figure 2: Number of comments in each class

all other trivial comments. We manually labelled the miscellaneous category from the creators' point of view.

As mentioned previously, some of the comments can belong to more than one class. For instance, the first sentence in "Your solution is not practical. Can you suggest another one?", suggests the negative class while the second sentence suggests the interrogative class. In such situations, we classified the comment based on the importance to the content creator. In this example, we assumed it as an interrogative sentence because it is more important to answer the question to increase odds of the viewer to return and stay engaged in the content of the channel. Fig. 2 shows the visual presentation of the quantity of comments in each class.

3.2 Data Pre-processing

It is important to clean the data and have them in appropriate format to improve classification. The data pre-processing step handles the following factors that make the classification process difficult:

(1) Non-standard language: The texts used in the comments section do not always employ standard English. Comments often contain slangs and improper form of words, making it difficult to extract features from them.
(2) Spelling errors: Commenters often do not pay attention to spellings due to the informal setting. Such spelling errors need to be corrected; otherwise, the words would add unnecessary features to the classification model, decreasing the overall accuracy of the classifier and impacting efficiency of the approach. For example, the words "plz" and "please" convey the same meaning in informal writing, but if the incorrect spelling is ignored, they would be treated as two different words by the classifier.
(3) Unformatted texts: These refer to comments containing computer codes. These do not contribute to the feature extraction accuracy; rather, they add unnecessary load to the feature matrix.
(4) Trivial comments: Not all the comments posted were about the video or related to the channel. A large number of viewers comment in order to market their products or just to show their presence. These comments are not useful to the content creators and only add unnecessary overhead.

Above issues are common in platforms like YouTube because of the informal nature of communication. We addressed these issues using the following pre-processing steps:

- lowercasing
- removing URLs
- removing new line character ("\n")
- removing punctuations
- removing integers
- removing emojis
- correcting spelling errors
- lemmatizing
- removing stopwords

Given the nature of this study, lowercasing was relevant because the same word would have been identified as a different feature had some letters been capitalized (for example, "Love" and "love" are two different words for a computer). The removal of URLs, new line characters, punctuation, integers, and emojis was performed because they did not provide useful information for feature extraction; rather adding unnecessary complexity to the model.

Since there are a lot of spelling errors in informal writing, we corrected the typos using the autocorrect-library from Python. We used Lemmatization to analyze the words morphologically and group the similar words together. Furthermore, there are certain frequent words that do not add significant meaning to the sentence such as "is", "are", and "it". They were removed from the corpus. However, stopwords were not removed from all the classes of comments because stopwords for one category might be important for another category. For example, "not" and "no" are important for the negative class whereas they are not important for other classes. Stopwords were used from nltk English corpus, which consists of 179 stopwords. Table. 3 shows the stopwords that were not considered in each category.

3.3 Converting text into features

Once pre-processing is completed, the pre-processed text is filtered so that only the necessary features remain in the corpus. This helps in reducing the load to the classifiers and also increases the accuracy of selected models [19].

The well-known techniques for vectorizing a corpus of text include document frequency, tf-idf vectorizer, hashing vectorizer, and Word2Vec. We selected document frequency vectorizer and tf-idf vectorizer for this paper. Using these two methods we can study the behaviour of different classification models under two different

Table 3: Stopwords ignored in each class

Class	Ignored Stopwords
Positive	NA
Negative	no, not
Interrogative	how, what, which, who, whom, why, do, is, does, are, was, were, will, am, are, could, would, should, can, did, does, do, had, have
Imperative	could, would, should, can
Corrective	NA
Miscellaneous	NA

conditions. Document frequency (df) vectorizer gives importance to the term that has higher frequency in the document; whereas, *tf-idf* can incorporate the terms that are rarely present in the document. Unlike hashing vectorizer, we can examine the text features which are important to the model using the vectors generated by *df* and *tf-idf*. For rare or out of vocabulary terms (which might be important to a model), *Word2Vec* can not create an ideal vector for them and it is difficult to interpret those vectors because of hidden layers.

When calculating document frequency (Eqn. 1), if the same term is present multiple times in a comment, then its additional counts are not considered. Also, the terms that appear in less than or equal to 5 comments are ignored because they do not add value to the features. In the same way, if any term appears in the majority of the comments, it does not add value to the feature because it is not the distinguishable feature for a class. These terms are likely already filtered by the stopwords removal process. However, we ensure that only terms that significantly add value to their comment's class are considered.

$$df = \frac{n}{N} \tag{1}$$

where, df denotes document frequency, n denotes number of documents in which the term appears, and N denotes total number of documents where document means comment.

After above steps, 2210 terms (features) were derived and scaled from 0 to 1 using a min-max scalar (normalization). We performed this because some of the machine learning models cannot handle large ranges of data. Doing so also helps in speeding up some of the calculations.

$$MinMaxScaler(x) = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{2}$$

where, x is observed value, x_{min} is the minimum value of that class and x_{max} is the maximum value of that class.

The second feature extraction technique used in this paper is *tf-idf* (term frequency - inverse document frequency). It not only considers the frequent terms, but also the rare terms.

$$tf\text{-}idf = tf * \log \frac{1}{df} \tag{3}$$

where, tf is the term frequency and df is the document frequency

For *td-idf*, we got 4304 features when both unigram and bigram were taken into account.

3.4 Model Selection and Hyperparameter Tuning

Each machine learning model has its own strengths and weaknesses when dealing with a text corpus. The fitness of a model for a specific dataset depends on the characteristics of the model as well as the features of the dataset. Since our text corpus is dense and has numerous features which significantly affects the classification task, the machine learning models selected are based on the density of the features and the number of classification categories (binary or multiple). Linear Support Vector Classification (Linear SVC), Logistic Regression, Multinomial Naive Bayes (Multinomial NB), Random Forest Classifier, and Decision Tree Classifier are selected as we work with a dense dataset and multiple classes. Especially, Random Forest Classifier helps prevent the overfitting problem and mitigates the impact of outliers, whereas Decision Tree Classifier can easily deal with the irrelevant features on the dataset.

For each model selected, the outcome can be enhanced by optimizing the hyperparameters. To understand how different parameters can significantly affect the performance of each model, we experimented all the models with the minimum subsets of those parameters (grid search strategy). Table. 4 shows all the values of the parameters that were tested for both df and $tf - idf$ measures. The columns "Best Value for" present the value with the best result, which were used for the final experiment.

4 RESULTS

4.1 Experimental Setup

All experiments were written in Python programming language and executed using Jupyter Notebooks. The "pandas" library was used for importing the data. The pre-processing step was carried out by regex functionality from the "re" library. Stopwords were removed by the "nltk" library. Similarly, spell correctors used the "autocorrect" library from Python. The feature selection and classification processes used the "sklearnmodule". For the training purpose, 80% of comments were randomly selected from the dataset, and the remaining 20% were used for testing. The experiments were run on Google Colab with 16.29 GB of RAM and 2.20GHz Intel(R) Xeon(R) CPU with GPU turned off. The overall process is shown in Fig. 3.

4.2 Result Analysis

We study the accuracy of results obtained from all the five models with two different feature extraction techniques in this section. The cross validation score ranged from 0.76 to 0.85, whereas F_1 score ranged from 0.81 to 0.87. Let us analyze the performance of

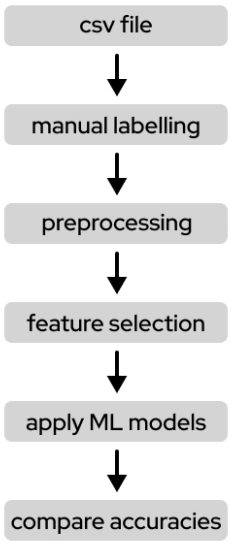

Figure 3: The complete classification process

Table 4: Parameters selected for tuning df and tf-idf features

Model	Parameters	Values Experimented	Best Value for *df*	Best Value for *tf-df*
Linear SVC	C	50, 10, 1.0, 0.1, 0.01	1.0	1.0
Logistic	C	100, 10, 1.0, 0.1, 0.01	1.0	10
Regression	solvers	newton-cg, lbfgs, liblinear	newton-cg	liblinear
	penalty	l1, l2, elasticnet, none	l2	l2
Multinomial NB	fit_prior	True, False	True	False
	alpha	0,0.5,1	1	1
Random Forest Classifier	n_estimators	10, 100, 1000	1000	1000
	max_features	sqrt, log2	log2	log2
	criterion	gini, entropy	entropy	entropy
Decision Tree Classifier	criterion	gini, entropy	gini	entropy
	max_features	sqrt, log2, auto, None	None	None

the classification models. For instance, we chose the results of the Random Forest classifier with document frequency as a feature selector. Some of the correctly classified comments for Random Forest classifier are shown in Table. 5 and the incorrectly classified samples are shown in Table. 6.

In Table. 5, the first comment had a distinguishing term "how" that implies interrogative class. Likewise, the second comment consisted of the term "Thank you" which suggested positive sentiment. Thus, these were correctly classified by the model.

The first two incorrectly classified comments in Table. 6 showed that the classifier gets confused when there are two types of sentences in the single comment. In the first comment, the first part of the sentence suggested imperative class, whereas the second part suggested a positive class. Earlier, when labelling the data, we classified the comment into imperative class because the content creator prioritized an imperative class over the positive class as they needed to focus on the grievances of the viewers (which was given by imperative class). Similarly, the third comment in Table. 6 was falsely classified. The most important word in this comment for distinguishing the category was "amazing". This word suggests the positive sentiment. Unfortunately, the classifier could not detect it because the corpus contained only a few positive sentences that included this term.

Table 5: Correctly classified comments

Index	Comment	Predicted Class
200	hi, how to do this environment setting on windows 10	interrogative
201	Thank you sooo much , sir!	positive
202	God has manifested to me in your form. Thank you.	positive
203	thank you sir helped alot.....	positive

Table 6: Incorrectly classified comments

Index	Comment	Predicted Class
133	Sir please teach these concepts using code...I really request you to explain it with code..Thank you so much for teaching all these concepts so we...	positive (imperative)
135	Time complexity for pure recursion should be pow(2, m+n). Correct me if I am wrong	miscellaneous (corrective)
213	You are an amazing Teacher...	miscellaneous (positive)
315	Your repeated the same meaning for about 1 minute	miscellaneous (negative)

The results of the best five models are shown in Table. 7 and Table. 8. We can see that all the five models performed satisfactorily except that Decision Tree Classifier showed weaker performance for both the feature selection methods. This is because of a high dimensional data (i.e. having dense features). This classifier splits the feature space at each node; as the feature space increases, the distance between the feature points also increase which makes it difficult to find a good split. Random Forest also works on the similar concept of Decision Tree, but it showed good performance because it does not use all the features rather it uses a collection of decision trees. This helps to reduce the distance between the the data points.

The reasonable cross validation scores for the models show that the model is not overfitted with the training data. Similarly, the spelling corrections performed on the training data had very insignificant impact on the performance of the models. The impact was only around 0.01%. Additionally, Fig. 4 and Fig. 5 show that the accuracy of each model increased with the increase in the data size.

Table 7: *df* used as a feature extraction method

Model Name	Cross Validation Score	F1-Score
Linear SVC	0.83	0.86
Logistic Regression	0.85	0.87
Multinomial NB	0.79	0.84
Random Forest Classifier	0.83	0.85
Decision Tree Classifier	0.80	0.83

Table 8: *tf-idf* used as a feature extraction method

Model Name	Cross Validation Score	F1-Score
Linear SVC	0.84	0.86
Logistic Regression	0.84	0.86
Multinomial NB	0.76	0.82
Random Forest Classifier	0.83	0.84
Decision Tree Classifier	0.80	0.81

As shown in Fig. 6 and Fig. 7, the poor performance in miscellaneous category can be inferred to the presence of diverse types of sentences that are classified under miscellaneous category, which confuse the models. The models perform relatively poor in imperative and negative categories as well. The imperative comments consisted of certain terms that fall under interrogative category, which complicated the classification task. For instance, "Would you please explain about library functions?" consisted of the terms "please", an imperative term; and "Would", an interrogative term. Likewise, the negative comments could contain both the interrogative part as well as the negative part. For example, "You are useless. Why do you keep repeating the same thing?". These types of comments create ambiguity, which result in degraded performance in the aforementioned categories.

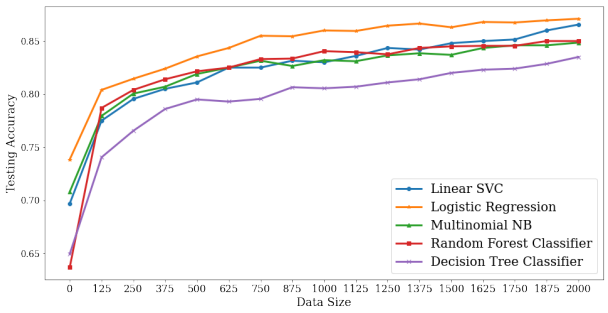

Figure 4: Testing Accuracy vs Data Size plot for five different models (df)

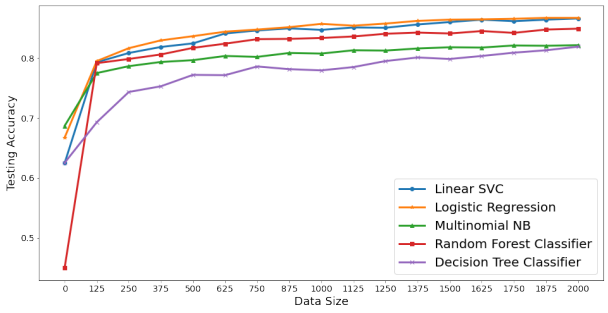

Figure 5: Testing Accuracy vs Data Size plot for five different models ($tf - idf$)

5 CONCLUSION

With the successful classification of the comments into their respective categories, a YouTuber can easily access each category of comment. The positive and negative category show the public sentiment on the video. Other categories aid the YouTuber to distinctly view the questions asked about the video and the suggestions provided to improve the content. This can help the YouTuber to avoid scrolling through hundreds of comments and filtering them manually for each video. While previous researchers focused either on sentiment analysis or classification of sentence of a niche, we have incorporated both the aspects. In this paper, we classified the comments using 5 different models on 2 feature selection methods. The experiments showed that best scores for cross validation and F_1 were obtained by Logistic Regression.

In future work, the number of classes and sub-classes can be increased to represent a more comprehensive comment classification. Likewise, the classification models and overall feature selection approach can be further improved for the comments that belong to more than one class. We also plan to extend this research by performing a comparative study with Explainable Neural Networks (xNNs) for comments classification.

REFERENCES

[1] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment Analysis of Twitter Data. In *Proceedings of the Workshop on Languages in Social Media* (Portland, Oregon) *(LSM '11)*. Association for Computational Linguistics, USA, 30–38.

[2] Rishabh Ahuja, Arun Solanki, and Anand Nayyar. 2019. Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor. In *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*. 263–268. https://doi.org/10.1109/CONFLUENCE.2019.8776969

[3] Khin Zezawar Aung and Nyein Nyein Myo. 2017. Sentiment analysis of students' comment using lexicon based approach. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*. 149–154. https://doi.org/10.1109/ICIS.2017.7959985

[4] Hanif Bhuiyan, Rajon Bardhan, and Dr. MD Rashedul Islam. 2017. Retrieving YouTube video by sentiment analysis on user comment. 474–478. https://doi.org/10.1109/ICSIPA.2017.8120658

[5] Zhalaing Cheung, Khanh Linh Phan, Ashesh Mahidadia, and Achim Hoffmann. 2005. Feature Extraction for Learning to Classify Questions. In *AI 2004: Advances in Artificial Intelligence*, Geoffrey I. Webb and Xinghuo Yu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1069–1075.

[6] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12, null (Nov. 2011), 2493–2537.

[7] A. Hassan and A. Mahmood. 2017. Deep learning for sentence classification. In *2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*. 1–5.

[8] Mohit Iyyer, Peter Enns, Jordan Boyd-Graber, and Philip Resnik. 2014. Political Ideology Detection Using Recursive Neural Networks. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, 1113–1122. https://doi.org/10.3115/v1/P14-1105

[9] Ammar Ismael Kadhim. 2019. Survey on supervised machine learning techniques for automatic text classification. *Artificial Intelligence Review* 52, 1 (2019), 273–292.

[10] Anthony Khoo, Yuval Marom, and David Albrecht. 2006. Experiments with Sentence Classification. In *Proceedings of the Australasian Language Technology Workshop 2006*. Sydney, Australia, 18–25. https://www.aclweb.org/anthology/U06-1005

[11] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[12] Amar Krishna, Joseph Zambreno, and Sandeep Krishnan. 2013. Polarity Trend Analysis of Public Sentiment on YouTube. In *Proceedings of the 19th International Conference on Management of Data* (Ahmedabad, India) *(COMAD '13)*. Computer Society of India, Mumbai, Maharashtra, IND, 125–128.

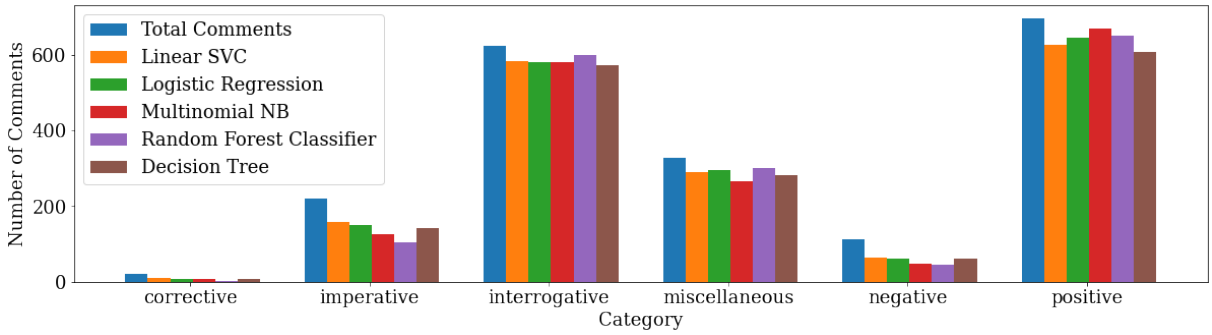

Figure 6: Performance of all the models in individual category for document frequency

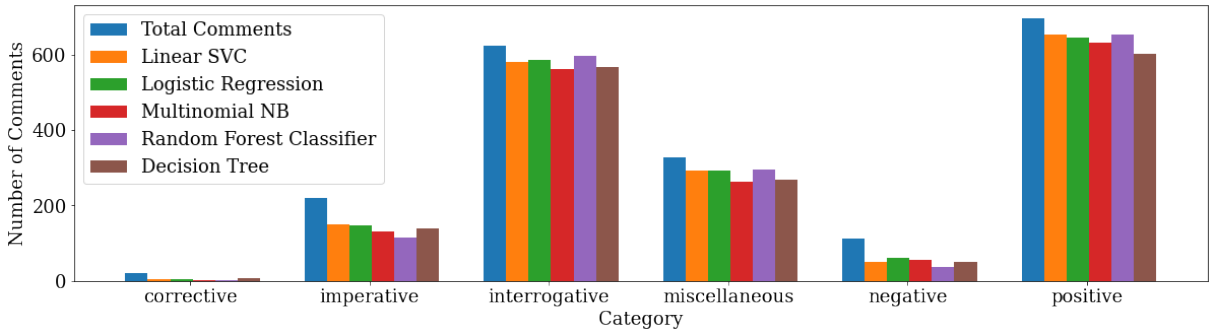

Figure 7: Performance of all the models in individual category for TF-IDF

[13] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. https://www.aclweb.org/anthology/P11-1015

[14] Amy Madden, Ian Ruthven, and David McMenemy. 2013. A classification scheme for content analyses of YouTube video comments. *Journal of Documentation* 69, 5 (2013), 693–714. https://doi.org/10.1108/JD-06-2012-0078

[15] Elizabeth Heidi Poche. 2017. *Analyzing User Comments On YouTube Coding Tutorial Videos*. Master of Science in Computer Science. Louisiana State University and Agricultural and Mechanical College, Baton Rouge, United States.

[16] Stefan Siersdorfer, Sergiu Chelaru, Wolfgang Nejdl, and Jose San Pedro. 2010. How Useful Are Your Comments? Analyzing and Predicting Youtube Comments and Comment Ratings. In *Proceedings of the 19th International Conference on World Wide Web* (Raleigh, North Carolina, USA) *(WWW '10)*. Association for Computing Machinery, New York, NY, USA, 891–900. https://doi.org/10.1145/1772690.1772781

[17] Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland, UK., 151–161. https://www.aclweb.org/anthology/D11-1014

[18] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-Based Methods for Sentiment Analysis. *Comput. Linguist.* 37, 2 (June 2011), 267–307. https://doi.org/10.1162/COLI_a_00049

[19] Yiming Yang and Jan O. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML '97)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 412–420.

[20] Douiji Yasmina, Mousannif Hajar, and Al Moatassime Hassan. 2016. Using YouTube Comments for Text-based Emotion Recognition. *Procedia Computer Science* 83 (2016), 292 – 299. https://doi.org/10.1016/j.procs.2016.04.128 The 7th International Conference on Ambient Systems, Networks and Technologies (ANT 2016) / The 6th International Conference on Sustainable Energy Information Technology (SEIT-2016) / Affiliated Workshops.

[21] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. A C-LSTM neural network for text classification. *arXiv preprint arXiv:1511.08630* (2015).