International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 05 | May 2022 www.irjet.net p-ISSN: 2395-0072

Popularity Prediction of video content over cloud-based CDNs using end-user interest

Burhanuddin Kanorwala[1], Siddharth Kothari[2], Saad Karol[3], Prof. Rupali Satpute

[1,2,3]Student, Dept. Electronics & Telecommunications, K. J. Somaiya Institute Of Engineering & Technology, Sion, Mumbai, India
Assistant Professor, Dept. Electronics & Telecommunications, K. J. Somaiya Institute Of Engineering &Technology, Sion, Mumbai, India

---

Abstract - *More is typically thought to be better, however, this is not the case when it comes to data. Every second, nearly a million minutes of video content crosses the network, and it will take an individual over 5 million years to watch all the videos across the Internet each month. We can't keep up with the tremendous growth of internet data because our capacity to handle, filter, and manage it isn't keeping up. CDNs have grown increasingly popular in today's world as the demand for faster and more consistent data access grows. However, storing all of the content on CDN servers has grown challenging. As a result, the goal of this project is to optimize video material stored in CDNs. In this project, we present a push-based caching technique for improving end-user quality of experience by locating relevant popular films in accordance with an area. Toclassifyfilmsaslow, medium, or very popular, a semi-supervised machine learning strategy was used. Popularity prediction of web content has exploded in popularity in recent years. However, there has been minimal research on forecasting video popularity based on pre-existing and significant video attributes. The experimental findings indicate high accuracy, validating the parameter choices and related processing.*

Key Words: Popularity prediction, Video content, CDNs

1. INTRODUCTION

A Content Delivery Network (CDN) is a collection of server nodes located around the world that are used to download resources (typically static content such as photos and JavaScript), resulting in faster delivery and lower latency. CDNs have grown in popularity throughout the years. Recent improvements in utility and cloud computing have made it possible to lease resources such as storage and bandwidth to create cloud-based Content Delivery Networks (CDNs). The petabytes of user-generated content traffic streaming over CDNs demonstrate the vastness of the content collection, making it critical to detect and select popular content for storage on the surrogate servers. The ability to predict video popularity is critical for the design and assessment of a CDN. The purpose of this research is to improve the effectiveness of CDN cache servers by using machine learning approaches to predict

video popularity. Prediction is the process of predicting the likelihood of a certain result on a specific dataset once an algorithm has been trained on it. For each tuple in the new data, the machine learning algorithm creates probable values for some unknown variable, allowing the model to determine what the most likely value of the unknown variable is for that case. Although current algorithms have been demonstrated to be useful for predicting video popularity, there are still a few unanswered concerns. One of the most important considerations is how much data is needed to train the model. More user behavior data or video correlation elements in general, according to research, contribute to greater algorithm performance.[9] Several YouTube measurement studies have looked at various statistical and behavioral features, but none have looked at popularity as thoroughly as we did. Recent works, [1]–[6] study distribution and temporal patterns of view count. Because consumers' attention and time are limited, it's not unusual that online video popularity is frequently distributed asymmetrically: a small number of videos acquire the bulk of views, while the rest of the videos are scarcely observed. [7] [8].

2. Methodology

To research, review and find and implement various methodologies for predicting the popularity of content to be put in cloud CDNs to

- Determine how the popularity of video content evolves based on this evolution description
- Seeking to estimate the popularity of video content in the future, by considering the current information available.
- Reduce the Caching server load.
- Increase hit ratio of content solicitation.
- Improve user experience by reducing response time. We have developed a three-phase approach.

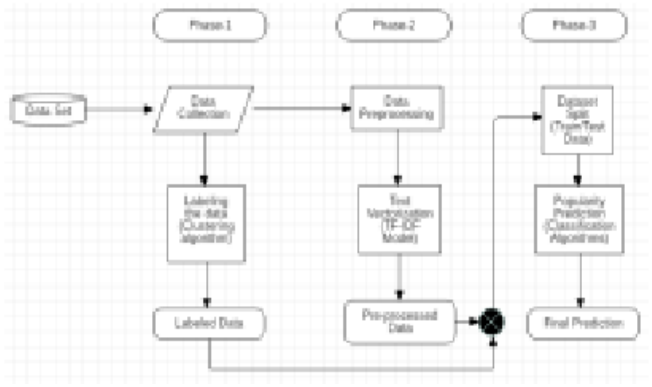

Chart -1: Flowchart describing the implementation process.

As shown in the figure, the implementation process is divided into three phases.

Phase-1: Data acquirement and labeling (Unsupervised Learning)

- The dataset used for this project – "Trending YouTube Video Statistics (Daily statistics for trending YouTube videos)" was acquired from Kaggle.com [10] (A Google-based community of data scientists and machine learning experts) which allows users to find and use published datasets for various data mining and machine learning tasks. This dataset was published by Mitchell J. in July 2019.

- The dataset had a totalofabout37000tuplesand16 features describing the video statistics – video id, trending date, title, channel title, categoryid, publish time, tags, views, likes, dislikes, comment count, thumbnail link, comments-disabled, ratings-disabled, video error or removed, description. The finalization of predictor variables is done in Phase-2.

- To predict the video popularity, there was no class available in the dataset, nor any attribute could act like it.

- The true popularity of a video can only be defined by the number of views on that video. It can be argued that likes and comments could also tell about how popular a video is, however, there are cases where disliked videos get millions of views and hence can be classified as popular.

- Therefore, considering the number of views on a video, the complete dataset was given to a K-MEANS clustering algorithm with the number of clusters as 3 (high, medium, and low).

- K-means Clustering is a vector quantization approach that divides n observations into k clusters (3 in this instance), with each observation belonging to the cluster with the closest mean(cluster center or cluster centroid), which serves as the cluster's prototype. Python's scikit-learn library was used.

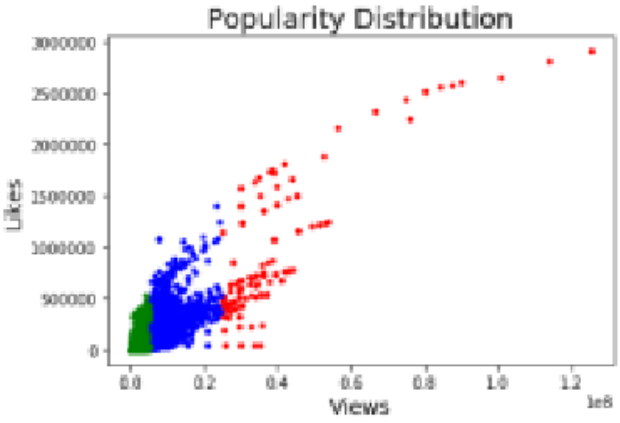

Fig -1: Figure 2: Popularity distribution trend create it.[11].

- The dataset is portioned into 3 classes, red – highly popular, blue – medium popular, and green – low popular videos. Hence, we are now equipped with a custom labeled dataset and will proceed to Phase-2.

3.3 Phase-2: Data Pre-processing

The first step toward dealing with large data is to identify and extract the correct features which will be used for further prediction purposes. It also reduces the amount of data that a machine must do computations on. From the earlier mentioned features, we selected a total of 8 features for popularity prediction which are –

1. Social features – likes, dislikes, comment count

2. Video features – video title, channel title, video tags, category id.

3. View count

Because the social features were numerical data, they could be simply passed to the machine learning algorithms, but because all of the video elements were text-based, they required a particular pre-processing strategy.

Step1: Punctuation removal – As YouTube tags are separated by the " |" symbol and also users can enter any

punctuation to make their video more appealing, all the punctuations were removed from all the 3 video parameters using the strings library of python3 [12].

Step 2: Foreign characters removal – Many videos contain alphabets and characters from languages other than English, hence those were removed for easy interpretation of textual features.

Step 3: Lowercase and tokenization: Any vectorization algorithm needs textual data in the form of tokenized words, and to reduce redundant words everything was converted to lowercase. Also, as channel title features contain mostly 2-3 words, hence they were merged since a channel title would remain consistent across the dataset.

After the basic pre-processing steps, we are equipped with clean consistent data which will now be passed to a TF-IDF vectorizer model. TF-IDF [13] stands for term frequency and inverse document frequency which is a numerical statistic intended to reflect how important a word is to a document. It was implemented using scikit-learn [11] Term frequency –

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

Equation 1.1: Term frequency
where,
$t_{i,j}$ - term frequency

$n_{i,j}$ – target word count in a sample

$\sum_k n_{i,j}$ – total target word count in the dataset.

Inverse document frequency –

$$idf(w) = \log \frac{N}{df_t}$$

Equation 1.2: Inverse Document Frequency
where,

$idf(w)$ - inverse document frequency

N - number of samples

df_t - number of samples having the target word.

The TF-IDF converts the textual columns into a vector having numerical value for each attribute. To use this vector as a single attribute for prediction purposes we assigned the textual features(title, channel title , and tags) as the summation of each row generated by the vector. Usually, the vector is directly passed into some machine learning algorithm but to use it as a single attribute we used the summation of attribute values for each column in the vector. Hence the textual features were dealt with and converted into numerical form with respect to the whole dataset. Now the text pre-processing and attribute normalization have been completed and we can move to Phase-3 of the project.

3.3 Phase-3: Applying Classification Algorithms (Supervised learning)

After the cleaned data has been acquired, we now move to perform machine learning algorithms on the dataset. The dataset was split as Training data – 80% and Testingdata – 20% using the scikit-learn library [11] in python and further the machine learning models are also used from the same.

A total of three machine learning algorithms were implemented – Naïve Bayes algorithm, Support Vector Machine (classifier) , and K-Nearest Neighbour algorithm.

1. NAÏVE BAYES ALGORITHM - A Naive Bayes classifier is a probabilistic machine learning model that is used for classification tasks. The crux of the classifier is based on the Bayes theorem [14].

$$P(c|X) = \frac{P(X|c).P(c)}{P(X)}$$

$$P(c|X) = P(X_1|c) \times P(X_2|c) \times P(X_3|c) \ldots . P(X_n|c) \times P(c)$$

Equation 2.1: Bayes Theorem
where,

c = hypothesis

X = evidence

$X_1 X_2 \ldots . X_n$ = attributes of the dataset

$P(c|X)$ – Posterior probability of hypothesis c given evidence X

$P(X|c)$ – Likelihood

$P(c)$ – Prior probability of hypothesis c

$P(X)$ – Prior probability of evidence

Precisely, Gaussian Naïve Bayes is used as the predictor features or the independent attributes are continuous in nature. Also, the Naïve Bayes algorithm assumes feature independence which is advantageous in this case as a video might become popular based on its content and irrespective of its attributes.

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Equation 2.2: Euclidian Distance

where,

x_i, y_i - coordinates of 2 points in i^{th} dimension

n – total number of dimensions.

2. SUPPORT VECTOR MACHINE – A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms. The objective of the support vector machine algorithm is to find a hyperplane in N-dimensional space (N — the number of features) that distinctly classifies the data points. It finds the best hyperplane that has the maximum margins between data points of all the classes [20]. Itisanelegantandpowerful algorithm for classification, and it is preferred for this project as the attributes are numerical in nature and could be optimally separated into three classes by three 7-dimensional (number of predictive features) hyperplanes.

3. K-NEAREST NEIGHBOUR – K-Nearest Neighbour is one of the most basic yet essential classification algorithms. It assumes that similar attributes appear in proximity, or similar things are close to each other[15]. It can use various methods for evaluating distances between points. We use the general Euclidean distance measure as there are many dimensions in the dataset.

K-Nearest Neighbours is used for this project as in contrast with the Naïve Bayes algorithm making attribute independence assumption, k-nn relies on the factor that closer points can be classified into a single class.

III. IMPLEMENTATION AND RESULTS

A. Accuracy Comparison

After the implementation of the selected approach, the results were observed.

K-Nearest Neighbour algorithm performed the best, achieving an accuracy score of about 99%, while comparative accuracies of Naïve Bayes (97.2%) and SVM(96.1%)were low, however, the proposed approach has resulted in an overall high-performance yielding approach.

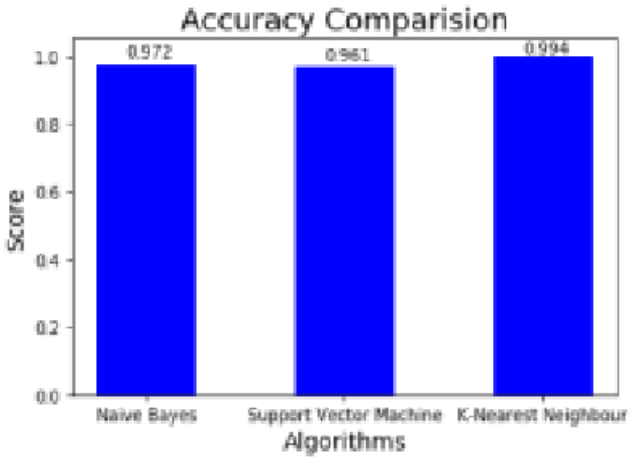

Figure 3: Accuracy Comparison chart

B. Confusion matrix evaluation

The K-nearest neighbor algorithm showed the best performance having only 9 misclassified examples. The Confusion Matrices of the three algorithms were also observed –

	Low Medium	High
Low	10489 262	8
Medium	0 396	9
High	0 7	35

Naïve Bayes classified a total of 286 cases incorrectly, where mostly Low popular videos were classified as Medium popular

	Low	Medium	High
Low	10759	0	0
Medium	308	97	0
High	34	0	8

Table 2: Confusion Matrix of SVM

The SVM was the most erroneous algorithm with a total of 342 misclassified instances where it was mostly unable to predict High popular videos.

	Low	Medium	High
Low	10759	0	0
Medium	5	400	0
High	0	4	38

Table 3: Confusion Matrix of K-Nearest Neighbor

C. Prediction time evaluation

Lastly, the prediction time for a raw tuple for each algorithm was examined, which had little variation as the time taken by Naïve Bayes was a maximum of 18.2 ms, and SVM and K-Nearest Neighbours were predicted in almost comparable time as shown below.

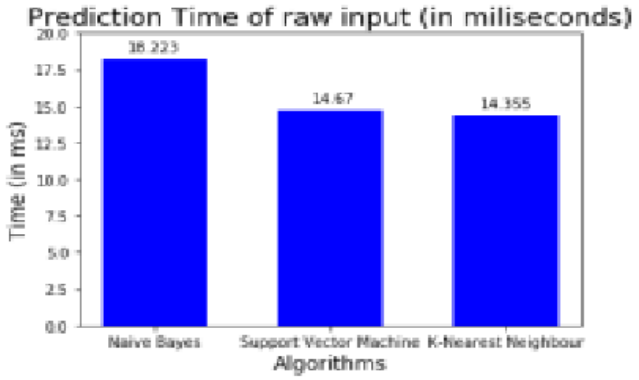

Figure 4: Prediction time analysis.

IV . FUTURE SCOPE

- This project uses a static historic dataset for prediction purposes, the same approach could be tested on a dynamic dataset where the social attributes keep hanging, so the model could handle dynamic data.
- The popularity of a video depends on its features to some extent but the most important factor in determining the popularity of the video would be the video content. Furthermore, image processing could be done on the video content to determine the actual context and make the prediction based on that.
- Additionally, foreign language characters could be translated to improve the overall performance.

V . CONCLUSION

CDN optimization is a big problem for us as CDNs become more prevalent by the day. We proposed and tested a prediction approach in this research that may be used to detect popular content in advance and subsequently pushed into a CDN server, justifying the use of a Push-based caching technique. We used efficient pre-processing algorithms to extract specific parameters that were both continuous and available at the time of video upload, and the TF-IDF vectorizer to deal with textual features. The textual qualities were given a score, which the machine learning method used as input parameters. We utilized three algorithms: Nave Bayes,K-Nearest Neighbour, and SVM classifiers, all of which produced high accuracy scores of up to 99percent (K-NN). Lastly, we compared the accurate and inaccurate predictions of each model and analyzed their prediction time for raw inputs.

VI. ACKNOWLEDGMENT

The authors gratefully acknowledge the teachers and friends for their support and guidance. The work would not have come to fruition without their supervision and selfless help. All the authors are in debt to Prof.Rupali Satpute for her guidance, without which the goal of the project would not have been realized.

VII. REFERENCES

1. M. Cha et al., “I tube, you tube, everybody tubes: Analyzing the world’s largest user-generated content video system”, in IMC ’07: Proc. of the 7th ACM SIGCOMM conference on Internet measurement, 2007.

2. X. Cheng, C. Dale, and J. Liu, “Statistics and social network of youtube videos”, IWQoS ’08: Proc. of the 16th International Workshop on Quality of Service, June 2008.

3. P. Gill et al., “Characterizing user sessions on YouTube”, MMCN 2008: Proc. of the 15th SPIE/ACM Multimedia Computing and Networking, 2008.

4. M. Zink et al., “Watch global,cachelocal:YouTubenetworktrafficat a campus network-measurements and implications”, MMCN 2008: Proc.ofthe 15th SPIE /ACM Multimedia Computing and Networking, 2008.

5. Phillipa Gill et al., “Youtube traffic characterization: a view from the edge”, in IMC ’07: Proc. of the 7th ACM SIGCOMM conference on Internet measurement, 2007.

6. F. Benevenuto et al., “Video interactions in online video social networks”, ACM Trans. Multimedia Comput.Commun.Appl.,vol.5, no. 4, 2009.

7. Cisco visual networking index: Forecast and methodology, 2014-2019,

8. F Wu and B. A Huberman, "Novelty and Collective Attention", Proceedings of National Academy of Sciences, 2007, 104(45): 17599-17601

9. Cha M, KwakH, RodriguezP,etal. I tube, youtube, every body tubes: analyzing the world's largest user generated content video system[C]//Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. ACM, 2007: 1-14.

10.Mitchell J, "Trending YouTube Video Statistics", Feb 2020
[Online].Available:https://www.kaggle.com/datasnaek/youtubenew

11.Learn, "scikit-learn machine learning in Python", May2020[Online]. Available:https://scikit-learn.org/stable/ 12.Python Documentation, String , "Common string operations", Apr 2020 [Online]. Available: https://docs.python.org/3/library/string.html
13.CoryMaklin,"TFIDF|TFIDFPythonExample",Apr2020,[Online].Available:https://towardsdatascience.com/natural-language-processing-fea ture-engineering-using-tf-idf-e8b9d00e7e76

14.Rohith Gandhi, "Naive Bayes Classifier", May 2020, [Online].Available:a.https://towardsdatascience.com/naive-bayes-classifier-81d512f5 0a7c

15.Onel Harrison, "Machine Learning Basics and K-NearestNeighbours Algorithm", May 2020, [Online]. Available: https://towardsdatascience.com/machine-learning-basics-with-th e-k-nearest-neighbors-algorithm-6a6e71d01761