Transactions on Computer Science and Intelligent Systems Research
ISSN: 2960-1800, eISSN: 2960-2238 | Volume 9
AIDML 2025

Comparing Machine Learning Models for Predicting YouTube Video Like-to-View Ratios

Xinming Zhao *

School of Jinan Jiaxuan School, Jinan, 250109, China

* Corresponding Author Email: outlook_C40FE0FE9A19369F@outlook.com

Abstract. With YouTube's dominance as a video-sharing platform, the like-to-view ratio (LVR) has emerged as a critical metric for quantifying audience engagement beyond raw view counts. Current approaches lack systematic model comparisons and actionable optimization strategies. This study bridges these gaps by rigorously evaluating tree-based ensembles (Random Forest, XGBoost, LightGBM) on raw metadata, demonstrating XGBoost's superior performance (R^2=0.8703) over both individual models and a Voting Regressor (R^2=0.8654); developing an operational framework that combines XGBoost predictions with LLM-generated suggestions (e.g., optimal publishing times); and deploying this system via a web tool for real-time decision support. The results reveal temporal features and hyperparameter tuning as key accuracy drivers. Despite excluding visual/audio elements due to data limitations, the implemented XGBoost-LLM hybrid system successfully bridges predictive analytics and creator guidance. Future work should explore multimodal data integration and cross-platform trends to enhance prediction robustness. This research advances video analytics by demonstrating how machine learning can optimize data-driven content optimization, offering both theoretical insights into model selection and practical tools for creators.

Keywords: YouTube video prediction; machine learning; ensemble learning.

1. Introduction

With over 2.5 billion monthly active users [1], YouTube has become the world's leading video-sharing platform [2], making the quantitative analysis of engagement metrics crucial for content creators, marketers, and recommendation systems. Among these metrics, the like-to-view ratio (LVR)—defined as the number of likes divided by the total number of views—serves as a normalized proxy for audience satisfaction and content quality [3]. Compared to absolute measures such as raw view counts, the LVR mitigates biases introduced by video virality or popularity skew [4], enabling fairer cross-video comparisons [5]. Prior studies have emphasized LVR's value as an evaluative tool, yet have often neglected systematic modeling efforts using modern ensemble learning [6]. Moreover, existing creator-facing tools tend to offer generic suggestions (e.g., "increase video duration"or "improve thumbnail design" [7]), lacking predictive customization and empirical grounding [8]. To address these gaps, this study proposes a machine learning framework for LVR prediction, leveraging tree-based ensemble models—Random Forest, XGBoost, and LightGBM—trained on raw video metadata. The top-performing model is then operationalized into a web-based decision support tool that integrates LVR prediction with actionable suggestions generated by a large language model (LLM), while accounting for potential technical debt [9]. This hybrid approach aims to bridge the divide between data-driven modeling and practical creator guidance, a challenge rarely addressed in existing research [10].

This paper presents a comprehensive evaluation of multiple ensembles learning models, all trained on an identical set of features, and includes a benchmark comparison with a Voting Regressor that combines predictions through weighted averaging. Subsequently, XGBoost emerges as the most accurate individual model (R^2 = 0.8703), outperforming both LightGBM (R^2 = 0.8694) and Random Forest (R^2 = 0.8213), as well as the aggregated Voting Regressor (R^2 = 0.8654), thereby questioning the presumed advantage of ensemble integration in this regression context. Building upon these findings, a fully functional web-based application is developed, incorporating the XGBoost model

alongside real-time outputs from a language model to deliver tailored recommendations aimed at optimizing video performance.

2. Methodology

2.1. Research process

The study's framework (Figure 1) progresses through four phases: data preprocessing; model training with Bayesian hyperparameter optimization; performance evaluation using regression metrics (RMSE, MAE, R^2); and tool deployment via a modular API-HTML architecture with latency under 500ms. This design ensures rigorous comparison while maintaining real-world applicability.

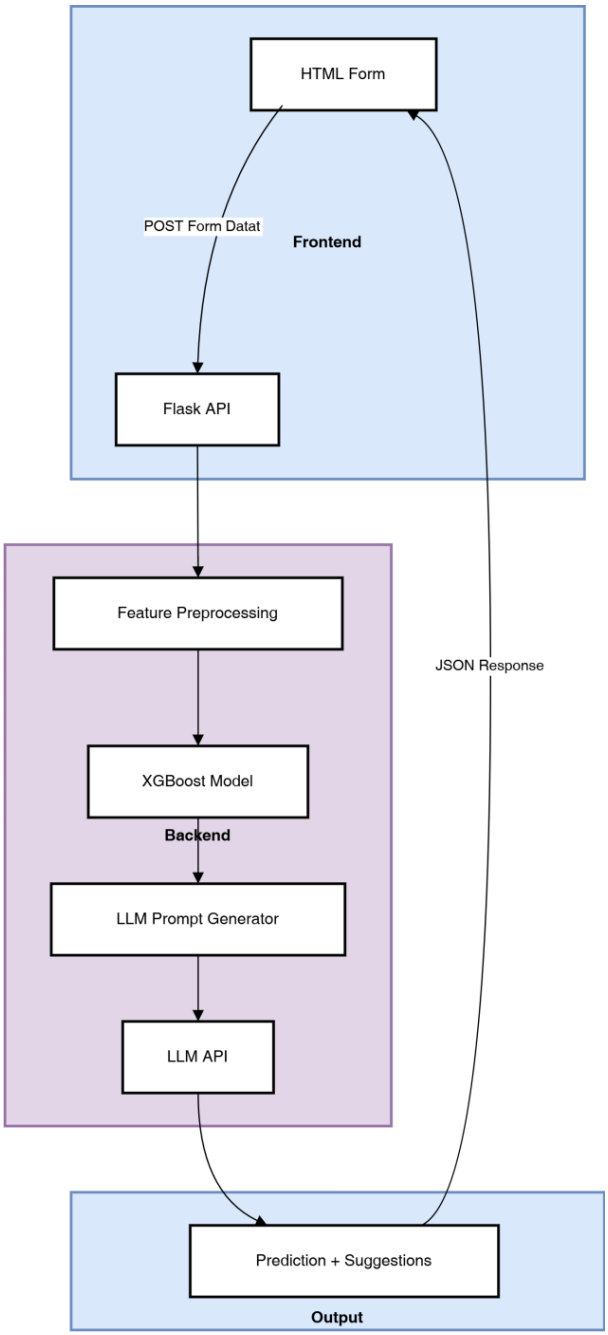

Figure 1. Framework for YouTube LVR prediction system (Picture credit: Original)

2.2. Dataset

The dataset used in this study contains 92,275 samples and 20 columns, including 19 input features and 1 target variable (the like-to-view_count ratio) according to Table 1. It is a multivariate tabular

dataset, with each row representing a YouTube video on a particular trending date. The dataset was originally released for a machine learning challenge, where the objective is to predict the target variable based on the video's metadata.

Table 1. YouTube Video Metadata Features and Descriptions

Feature Name	Description
Video identifiers (id, video_id)	Each entry contains unique identifiers combining video IDs and trending dates, allowing for precise tracking of video performance over time.
Textual information (title, description, tags)	This category includes all text-based content features that may influence viewer engagement, comprising video titles, detailed descriptions, and topic-related tags.
Temporal attributes (publishedAt, trending_date)	Timing-related features capture both the original publication date and subsequent trending dates, enabling analysis of temporal patterns in video popularity.
Channel information (channelId, channelTitle)	Metadata about content creators includes unique channel identifiers and names, permitting investigation of channel-specific effects on engagement.
Engagement metrics (view_count, likes, dislikes, comment_count)	While these key performance indicators are available in the training data (except for the test set), they provide crucial ground truth for model development. Notably, the target variable (like-to-view ratio) is derived from these metrics.
Platform settings (comments_disabled, ratings_disabled)	Binary flags indicate whether YouTube's interactive features were disabled for each video, which may significantly impact engagement patterns.
Visual indicators (thumbnail_link, has_thumbnail)	This group contains both thumbnail image links and availability indicators, offering the potential for visual content analysis through computer vision approaches.
Video duration (duration_seconds)	The length of each video in seconds serves as an important content characteristic that may affect viewer engagement and retention.

2.3. Data Preprocessing

To ensure the quality and consistency of the input features, this paper performed a series of preprocessing steps on the training and test datasets.

First, irrelevant features including view_count, likes, dislikes, description, and unique identifiers (video_id, thumbnail_link, id) were removed to reduce noise and dimensionality.

Second, temporal features were extracted from the published At column to capture time patterns, deriving month, hour, Dayo week, and monthsold features. The hour feature was further categorized into four intervals (Night, Morning, Noon, Evening) to better represent daily patterns, after which the original trending date and monthsold columns were dropped.

Third, categorical and boolean features underwent encoding transformations. Boolean values were converted to integers (0/1), while categorical features including channel Title, channeled, title, and tags were encoded using frequency encoding for high-cardinality features and label encoding for others.

Fourth, missing values in the duration_seconds column were addressed by imputing with the median value to maintain robustness against outliers.

Finally, all numerical features were standardized using StandardAero to ensure consistent feature scales and improve model convergence.

2.4. Model Overview

2.4.1. Basic Model.

The Random Forest Regressor is an ensemble learning method that combines multiple decision trees and averages their predictions for regression tasks. By training each tree on a random subset of the data and features, this approach reduces overfitting and enhances generalization performance compared to a single decision tree.

For the model parameters, the number of trees in the forest was set to 300, with a maximum depth of 20 for each individual tree. The minimum number of samples required to split an internal node was 5, while the minimum number of samples required at a leaf node was 2. To ensure reproducibility, a fixed random seed of 0 was used.

The model was trained on the training dataset (x_train) and corresponding target values (y_train) using the specified parameters. This configuration aims to balance predictive accuracy and robustness while preventing overfitting.

The XGBoost Regressor is a gradient boosting algorithm designed for structured or tabular data. Unlike random forests, which train trees independently, XGBoost builds an ensemble of decision trees sequentially, with each new tree correcting errors from the previous ones. This approach, combined with advanced regularization techniques, has made XGBoost a top-performing method in machine learning competitions due to its computational efficiency and predictive accuracy.

For model tuning, key hyperparameters were optimized using Bayesian Optimization. The number of boosting rounds (trees) was set within a range of 50 to 300, while the maximum depth of each tree varied between 3 and 10 layers. The learning rate, controlling the step size during loss minimization, was tuned between 0.01 and 0.3. To mitigate overfitting, subsampling was applied by randomly selecting 50% to 100% of training samples for each tree, and column subsampling restricted feature usage to 50% to 100% per tree. Early stopping was implemented with a patience of 20 rounds, halting training if validation performance failed to improve, thereby conserving computational resources. A fixed random state of 0 ensured reproducibility across experiments.

The Light Gradient Boosting Machine (LightGBM) is a gradient boosting framework that prioritizes speed and efficiency in model training. Unlike traditional level-wise tree growth methods, LightGBM adopts a leaf-wise expansion strategy, which often leads to faster convergence and improved model accuracy. This makes it particularly suitable for handling large datasets and competitive machine learning scenarios where both performance and computational efficiency are critical.

For hyperparameter optimization, Bayesian Optimization was employed to determine the ideal settings. The number of leaves per tree was tuned within a range of 20 to 80, while the maximum tree depth varied between 3 and 10 layers. The learning rate, which controls the step size during gradient updates, was optimized between 0.01 and 0.3. To enhance generalization, subsampling randomly selected 50% to 100% of training instances for each tree, and column subsampling restricted feature usage to 50% to 100% per split. The model used 1000 boosting rounds (trees) and incorporated early stopping with patience of 20 rounds to terminate training if validation performance plateaued, thereby preventing overfitting and unnecessary computation. A fixed random state of 0 ensured reproducible results across training runs.

2.4.2. Integrated Model.

The Voting Regressor is an ensemble method that combines multiple regression models to improve prediction accuracy. It aggregates the predictions of various base models (in this case, Random Forest, XGBoost, and LightGBM) through a simple voting mechanism to produce a final output. This method leverages the strengths of each individual model, potentially reducing bias and variance.

The Voting Regressor was trained using the previously defined models (rf_regressor, best_xgb, and best_lgb). Each model contributes to the final prediction, and the Voting Regressor aggregates the predictions to make the final output.

3. Experiment Results and Analysis

3.1. Model Evaluation Method

Cross-validation and scatter plots for predicted vs. actual values were used to evaluate the performance of the models, and performance metrics such as R^2, MAE, MSE, and RMSE were calculated. Both train and test MAE were computed to assess whether the model is overfitting by comparing the two values.

The R^2 score measures the proportion of the variance in the target variable that is explained by the model. A higher R^2 score indicates a better model fit. MAE measures the average magnitude of the errors in the predictions, with smaller values indicating better performance. MSE reflects the squared errors of predictions, with smaller values being better. RMSE is the square root of MSE, which provides an error metric in the same units as the target variable.

3.2. Experiment Results Comparison

The dataset was divided into input features (x) and the target variable (y). The data was then split into training and testing sets (70% training, 30% testing) randomly to evaluate the model's performance on unseen data. Each regression model underwent hyperparameter tuning via Bayesian optimization.

Table 2. Model Evaluation Results

Model	R2 Score	MAE Train	MAE Test	MSE	RMSE
Random Forest	0.8213	0.0086	0.0102	0.0003	0.0172
XGBoost	0.8703	0.0059	0.0078	0.0002	0.0146
LightGBM	0.8694	0.0058	0.0078	0.0002	0.0147
Voting egressor	0.8654	0.0064	0.0082	0.0002	0.0149

As shown in Table 2, XGBoost has the highest R^2 score of 0.8703, meaning it can explain 87% of the variance in the data, performing the best among all models. LightGBM follows closely with an R^2 score of 0.8694, which is nearly identical to XGBoost's performance. In comparison, Random Forest has the lowest R^2 score of 0.8213, suggesting that it has a relatively weaker fit to the data. Voting Regressor has an R^2 score of 0.8654, which, while lower than XGBoost and LightGBM, still demonstrates strong predictive performance.

As shown in Table 2, XGBoost and LightGBM achieve the lowest MAE on both training (0.0059 and 0.0058) and test sets (both 0.0078), indicating consistent and accurate performance without signs of overfitting. On the other hand, Random Forest has a higher MAE of 0.0102 on the test set, indicating relatively larger prediction errors. Voting Regressor has an MAE of 0.0064 on the training set and 0.0082 on the test set, slightly higher than XGBoost and LightGBM, but still provides competitive results.

As shown in Table 2, XGBoost, LightGBM, and Voting Regressor all have an MSE of 0.0002, indicating the best performance. In comparison, Random Forest has a slightly higher MSE of 0.0003, suggesting that its predictions have slightly larger errors.

From Table 2, we see that XGBoost and LightGBM have the lowest RMSE values of 0.0146 and 0.0147, respectively, indicating the smallest prediction errors. Random Forest has a higher RMSE of 0.0172, suggesting that its predictions are more error-prone. Voting Regressor has an RMSE of 0.0149, slightly higher than XGBoost and LightGBM, but still demonstrates robust performance.

3.3. Model Performance Comparison

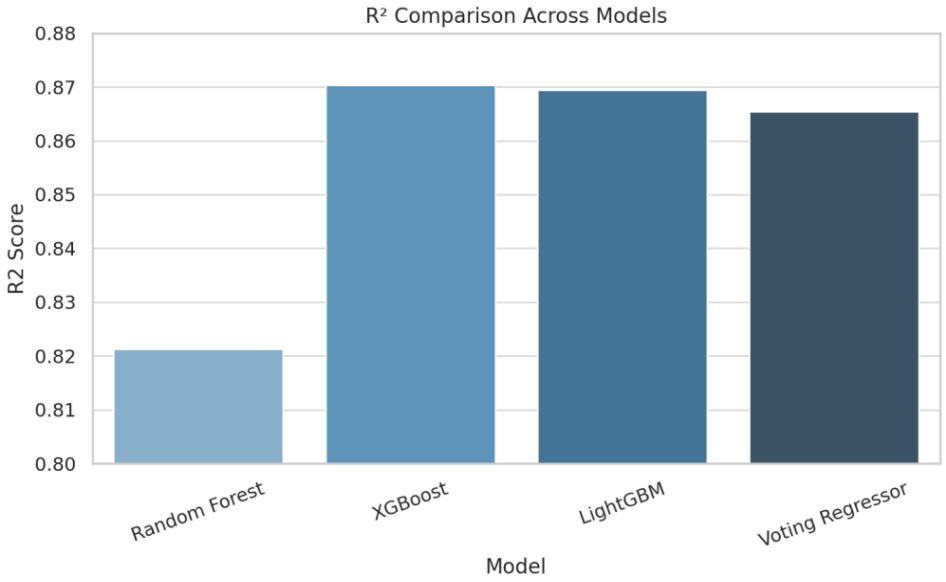

Figure 2. R² Comparison Across Models (Picture credit: Original)

As shown in Figure 2, XGBoost and LightGBM outperform other models in R², with XGBoost reaching the highest R² score of 0.8703. Random Forest shows a relatively weaker performance, with a lower R² score of 0.8213, indicating it explains less of the variance in the target variable. However, it remains a solid model with good predictive power. Voting Regressor, combining multiple models, performs similarly to XGBoost and LightGBM, with only minor differences in MAE and RMSE.

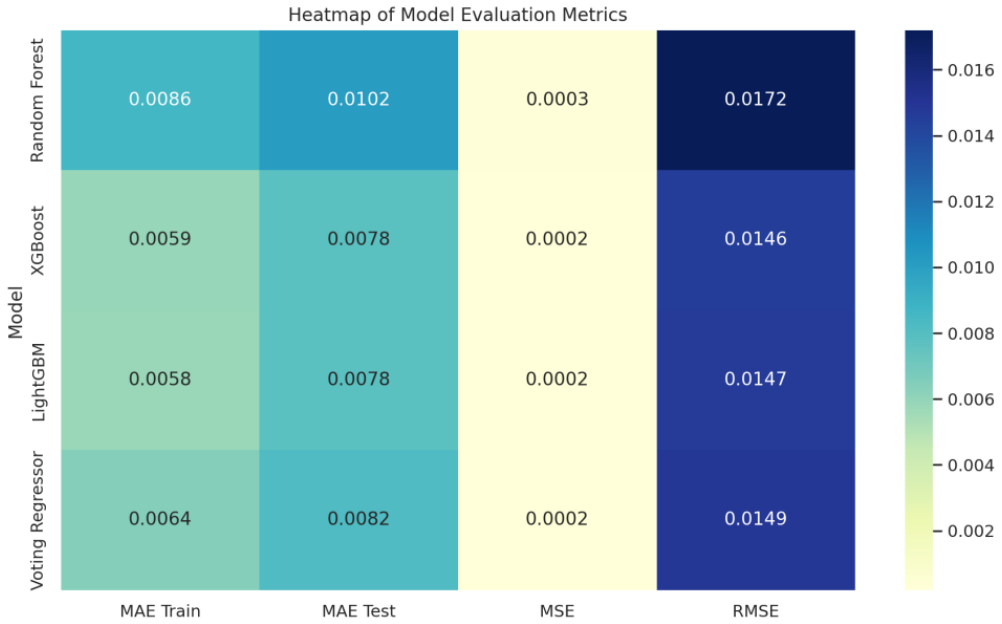

Figure 3. Heatmap of Model Evaluation Metrics (Picture credit: Original)

Figure 3 shows the heatmap of the models' performance metrics. While XGBoost and LightGBM lead across most metrics, particularly R², MAE, and RMSE, Random Forest exhibits higher MAE and RMSE, reflecting larger prediction errors. The Voting Regressor maintains competitive performance with slightly higher MAE but still offers robust and reliable predictions.

3.4. LLM-Powered API Interaction Layer

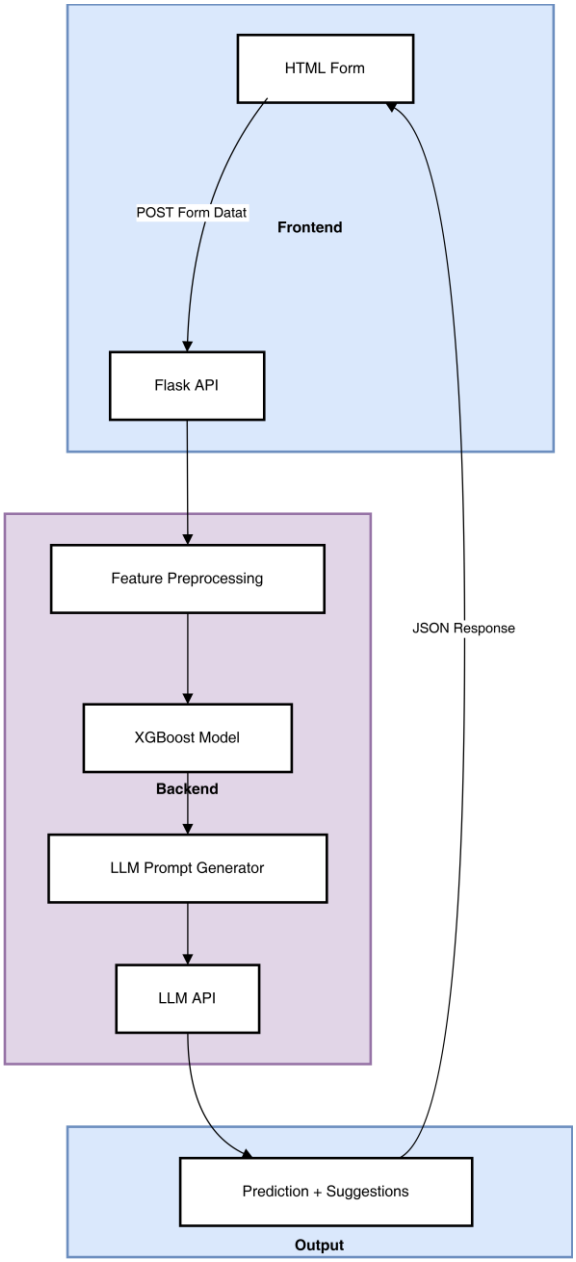

Figure 4. The Architecture Diagram of LLM-Powered API Interaction Layer (Picture credit: Original)

In order to better reflect the practical application of the model, this paper deploy a lightweight web application. As shown in Figure 4, It includes the Frontend Interface, which is an HTML/CSS/JavaScript interface (index.html) that collects user inputs including video metadata (title, description, etc.) through form submission. Backend API, A Flask-based Python server (app.py) handling HTTP requests. It works for Preprocesses input features (e.g., text cleaning, duration normalization); loading the pre-trained XGBoost model for LVR prediction and generating customized suggestions using LLM prompts based on prediction results. Also, Model Serving is included, which loads the best prediction model XGBoost obtained in previous research into memory for real-time inference.

4. Conclusion

This study identifies the most effective machine learning model for predicting YouTube's like-to-view ratio and introduces a hybrid system that integrates predictive modeling with guidance

generation. Experimental results indicate that XGBoost demonstrates superior performance, exceeding that of LightGBM, Random Forest, and even a weighted ensemble Voting Regressor. This suggests that combining similar models may not always yield additional advantages in regression tasks. The final system integrates real-time predictions with dynamic, context-aware suggestions, offering a practical tool for content optimization.

Several limitations were observed during the development process. Key visual features, such as thumbnails, were not included due to preprocessing constraints, even though they are known to influence user engagement significantly. Similarly, audio characteristics like background music or volume dynamics were excluded as they were not available in the metadata. Moreover, the system does not currently account for external traffic influences, such as short-form content on other platforms driving views to YouTube, which could introduce additional variability.

To further improve both accuracy and recommendation quality, future work will explore integrating heterogeneous model types, including neural network architectures like tabular transformers or deep ensemble frameworks. Expanding the input space to include multimodal data—such as images, audio, and text—as well as incorporating user feedback mechanisms and cross-platform datasets, will further enhance the model's capacity to better support video creators in optimizing engagement while minimizing operational overhead.

References

[1] Statista. YouTube: Global Active Users 2023. Statista, 2023.

[2] Covington P, Adams J, Sargin E. Deep Neural Networks for YouTube Recommendations. Proceedings of the 10th ACM Conference on Recommender Systems, 2016: 191 – 198.

[3] Sikdar S, Chaudhary A, Kumar S, Ganguly N, Chakraborty A, Kumar G, Patil A, Mukherjee A. Identifying and Characterizing Sleeping Beauties on YouTube. Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work and Social Computing Companion. New York: ACM, 2016: 405 – 408.

[4] Fu W W, Sim C. Aggregate Bandwagon Effect on Online Videos' Viewership: Value Uncertainty, Popularity Cues, and Heuristics. Journal of the American Society for Information Science and Technology, 2011, 62 (12): 2382 – 2395.

[5] Abisheva A, Garimella V R K, Garcia D, Weber I. Who Watches (and Shares) What on YouTube? And when? Using Twitter to Understand YouTube Viewership. Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM '14). New York: ACM, 2014: 593 – 602.

[6] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), 2016: 785 – 794.

[7] Song Y, Redi M, Vallmitjana J, Jaimes A. To Click or Not to Click: Automatic Selection of Beautiful Thumbnails from Videos. Proceedings of the 25th ACM International Conference on Information and Knowledge Management (CIKM '16). New York: ACM, 2016: 659 – 668.

[8] Koh B W, Cui F. Give a Gist: The Impact of Thumbnails on the View-through of Videos. SSRN Electronic Journal, 2020.

[9] Sculley D, Holt G, Golovin D, Davydov E, Phillips T, Ebner D, et al. Hidden Technical Debt in Machine Learning Systems. Advances in Neural Information Processing Systems, 2015, 28: 1 – 15.

[10] Liu J, Wang Y, Lyu Y, Su Y, Niu S, Xu X, Zhang Y. Harnessing LLMs for Automated Video Content Analysis: An Exploratory Workflow of Short Videos on Depression. Companion Publication of the 2024 Conference on Computer-Supported Cooperative Work and Social Computing. New York: ACM, 2024: 190 – 196.