Media Recommendation

Zongxian Feng
University of Illinois at Urbana-Champaign
zfeng13@illinois.edu

Chenxin Wang
University of Illinois at Urbana-Champaign
cw110@illinois.edu

Andres Suazo
University of Illinois at Urbana-Champaign
suazo2@illinois.edu

ABSTRACT

This report presents the development of a Chrome browser extension designed to enhance video discovery on YouTube by dynamically generating keyword recommendations based on user interactions. The extension employs natural language processing (NLP) techniques, specifically focusing on attention mechanisms, to analyze text interactions in real-time and tailor recommendations to the user's current engagement.

1 INTRODUCTION

YouTube, the leader in digital video broadcasting, receives an overwhelming 500 hours of video uploads every minute. This enormous volume presents significant discoverability challenges, leading to user overload and reduced content engagement. Unlike traditional YouTube recommendation systems that mainly rely on user history and predefined metadata, our project introduces a dynamic, real-time solution. This Chrome extension enhances user experience by leveraging advanced natural language processing (NLP) to dynamically generate keyword recommendations. Specifically, it analyzes user interactions within the comments section of videos, thus facilitating a more responsive and engaging discovery process.

Innovative search functionalities, similar to those seen in TikTok, enhance user engagement through interactive features such as hyperlinked keywords in comments [6]. For example, when users scroll through comments, our system identifies and extracts key phrases from the visible text and immediately suggests related videos. This mechanism not only promotes interactivity but also utilizes real-time, user-generated content to tailor search results, making the discovery process both interactive and user-driven.

This shift towards more dynamic and interactive content discovery tools is crucial as platforms like TikTok reshape how younger audiences search for information. Google's Senior Vice President of Google Knowledge and Information, Prabhakar Raghavan, has pointed out that many young people now prefer social media platforms over traditional search engines for casual inquiries, including finding places to eat [6]. Inspired by this trend, our extension aims to integrate similar functionalities into YouTube. By adapting recommendations based on real-time user interactions within the comment streams, our tool seeks to significantly enhance user engagement and satisfaction, bridging the gap between static video content and a lively, interactive search experience.

2 MOTIVATION

The motivation for developing this media recommendation extension comes from the need to address the issue of content availability and improve content discovery for YouTube's recommendation system.

The issue of content availability refers to difficulty caused by the vast quantity of content, making users hard to discover new and relevant content and often leading to choice overload. Creators also face challenges in gaining visibility and standing out. Discoverability becomes a key hurdle and forces creators to focus on click conversion rather than high quality.

YouTube's recommendation system lacks a mechanism to reflect users' interaction with current content in real time. This gap highlights the need for a system that can dynamically adjust recommendations based on immediate user feedback, such as likes and comments, to provide a more responsive and engaging user experience.

We proposed this Chrome browser extension called Media Recommendation in response to these challenges by providing personalized keyword recommendations that adapt in real-time to user behavior. This adopted dynamic approach is crucial in a digital landscape where users expect highly personalized experiences. It not only enhances user engagement by making content discovery more intuitive but also increases the visibility of creators' content by aligning recommendations with viewers' immediate interests.

3 RELATED WORK

The field of automatic keyphrase extraction has advanced significantly with the development of various methods, each tailored to improve how textual content is analyzed and understood in terms of extracting meaningful phrases that define the core topics of the text. Our project evaluated several state-of-the-art algorithms before selecting one that best meets the dynamic needs of a YouTube content discovery extension.

RAKE (Rapid Automatic Keyword Extraction) is an older, yet efficient method that identifies key phrases in text by analyzing the frequency and co-occurrence of words within a fixed window. RAKE benefits from simplicity and speed but lacks the semantic understanding that deep learning models provide, which is critical for interpreting the complex user interactions typical on YouTube [7].

YAKE (Yet Another Keyword Extractor) represents an improvement over traditional methods by considering multiple local features of words such as word position, word frequency, and word relatedness to contextually relevant neighbors. YAKE's lightweight nature and adaptability to single documents make it effective for real-time applications. However, it may still miss nuanced meanings compared to deep learning approaches [3].

KeyBERT uses a minimalistic approach by employing BERT embeddings to extract keywords that are semantically close to the document's overall context. While effective in capturing the essence of the document, KeyBERT struggles with real-time adaptability and can be computationally intensive for live applications [1].

DistilBERT and BERT Uncased models represent two variations of BERT optimized for efficiency and broader applicability. DistilBERT provides a streamlined version of BERT with reduced complexity, which offers faster processing times at the cost of some performance. The BERT Uncased model, processing text in a lowercase format, offers robustness in understanding the semantic content of text but lacks the finer differentiation capabilities provided by case-sensitive analysis. Both models, while powerful, do not specifically address real-time interaction or the extraction of context-specific keyphrases in dynamic environments like YouTube [2].

KBIR (Keyphrase Boundary Infilling with Replacement), our chosen model, builds upon the robustness of RoBERTa and introduces innovative features tailored for dynamic and context-sensitive applications. KBIR's Infilling Head and Replacement Classification Head provide it with a unique ability to adapt in real-time to new user-generated content [5]. This makes KBIR particularly suited for applications where user interaction frequently introduces new terminology and topics, such as on YouTube. Its performance in dynamically adjusting to contextual shifts without manual retraining surpasses that of the models discussed, making it the optimal choice for our project.

These considerations highlight the importance of selecting a model not only based on its current capabilities but also on its adaptability to the specific requirements of the deployment environment. KBIR's superior performance in handling real-time data and its ability to dynamically adapt to the nuances of user-generated content made it the standout choice for our application.

4 INTENDED USERS

The target users can be categorized into two main groups:

Content Viewers: Casual users of the platform who find themselves navigating through the immense amount of content available. They would benefit from a more tailored experience by having the tool recommend content that matches the viewer's interests or even aid in the discovery of new areas that might be of interest.

Content Creators: Owners of the content that is being distributed by the platform. Insights into viewer preferences and tendencies provide creators with a better way to engage with their audience as they would produce new content more aligned with the current interests of the audience.

5 MAJOR FUNCTIONS

The major function of the system is to deliver tailored recommendations that are highly personalized and context-aware.

The system is designed to analyze the current page metadata, understand the context of user interactions, and observe user actions such as scrolling and other forms of engagement within the comment section. By doing so, we can provide recommendations that are not only relevant to the content being viewed but also aligned with the user's active interests and behaviors.

6 IMPLEMENTATION

Figure 1 shows the architecture of the system. The Media Recommendation system provides a user-friendly interface in Chrome extension to display dynamically generated keyword recommendations and associated video information based on real-time analysis of user interactions. The implementation details are described as follows.

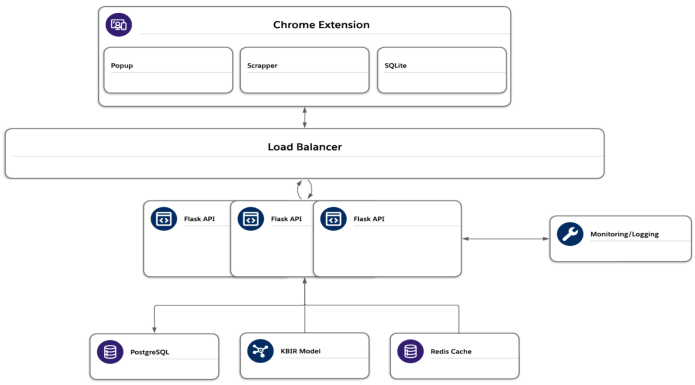

Figure 1: Architecture

6.1 Frontend

The frontend of the system is designed as a Chrome extension and implemented in HTML, CSS and JavaScript. The frontend mainly extracts comments from the current YouTube video page based on user interactions and sends them to the backend, enabling users to view a curated list of backend-generated recommendations.

The frontend retrieves the top five comments by default, ensuring that users are presented with the most relevant and engaging content initially. The extension adds a scroll event listener on the current page. As users scroll through the comments, the extension will dynamically extract comments within the viewport. This real-time feature ensures that the user's experience is seamless and that they are continuously presented with content that is currently in focus.

The frontend alsos add a click event listener on the current page to track user interactions within the comment sections such as clicking the like button, this allows the extension to extract comments that are liked by the user. It will then place a high emphasis on those comments, indicating a preference for those particular opinions or insights. This user-centric design allows for a more tailored experience, as the system learns from user behavior to prioritize and recommend comments that align with their interests.

6.2 Backend

The backend of the system is architected as a WSGI web application utilizing Flask and implemented in Python, which serves as a critical interface between the frontend and the model. The backend is designed to handle HTTP requests from the frontend, process the incoming video comments, and communicate with the external YouTube API to enhance the functionality of the system.

When receiving the video comment in raw string from the frontend, the backend performs a series of normalization steps before it is fed into the machine learning model. Normalization includes tasks such as trimming whitespace, removing special characters,

and tokenizing the string which allow the model to interpret the data accurately.

The backend is also responsible for interfacing with the YouTube API. For each keyword generated by the model, the backend queries the YouTube API to fetch the most relevant video and its associated metadata. Once the relevant video information is retrieved, the backend bundles the keyword and video data together and sends this packaged information back to the frontend. This bundled data enables the frontend to present users with enriched content that is directly related to the keywords of interest, thereby enhancing the user experience by providing contextually relevant recommendations.

6.3 Model

The model component is based on the KBIR approach, which enhances traditional keyphrase extraction techniques by incorporating two innovative heads: the Infilling Head and the Replacement Classification Head.

- **Infilling Head**: This component is crucial for enhancing the model's understanding of context within texts. It works by predicting missing information, which helps the model grasp the implicit meanings and connections in user-generated content. For example, in the sentence "Artificial intelligence has transformed industries from healthcare to finance. [MASK] can analyze vast amounts of data quickly, leading to more informed decisions," the model learns to infer suitable phrases to fill in the masked section, such as "It" referring to "Artificial intelligence", enhancing its contextual awareness beyond simple word-level predictions.
- **Replacement Classification Head**: This head trains the model to evaluate whether a phrase in a sentence can be replaced without losing the original meaning, which is pivotal for understanding the relevance and importance of terms within the user interactions. For instance, considering the replacement of "Artificial intelligence" with "Advanced technology" in the sentence "Artificial intelligence has transformed industries from healthcare to finance," the model assesses if the new phrase preserves the original sentence's intent and contextual meaning.
- **BIO Tagging in NER (Named Entity Recognition)**: Our selected KBIR model is fine-tuned on INSPEC dataset which employs BIO tagging, a common approach in NER tasks, where 'B' stands for the beginning of an entity, 'I' for inside, and 'O' for outside. This technique aids the model in distinguishing between different parts of data entries and better understanding their boundaries and relationships, essential for effective keyword extraction and replacement tasks.[4]

This layered approach allows KBIR not only to perform standard keyphrase extraction but also to adapt dynamically to the nuances of user-generated content, making it highly effective for real-time applications like the YouTube Chrome extension.

7 EVALUATION

Figure 2 shows the recommendation result in our system. In order to evaluate the effectiveness of our system, we implemented A/B testing as the primary method with the goal of determining if

Figure 2: Recommendation Result

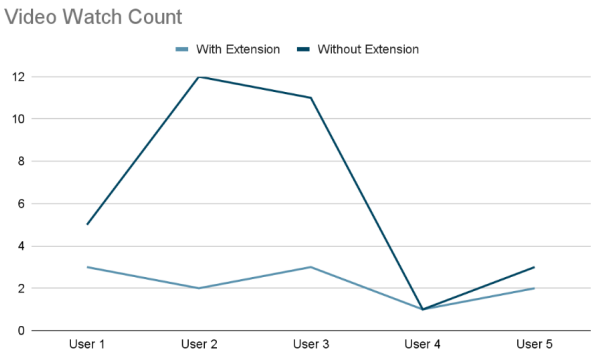

Figure 3: A/B Testing Result

our system resulted in increased user interactions with the search and higher engagement with recommended content compared to YouTube's standard recommendations.

We created two distinct user groups: one with the extension enabled and the other without it. By comparing the interaction metrics—such as likes, comments, and shares—and the duration of platform engagement between these two groups, we found that our extension will lead to a higher interaction count, reflecting increased user engagement, and consequently, an extended use time on the platform. The results of this A/B testing would provide

valuable insights into the extension's effectiveness in enhancing user experience and engagement, guiding further development and optimization efforts.

8 USER GUIDE

8.1 Installation

- Clone the repository: `git clone https://github.com/afesuazo/CS510_Project.git`
- CD to the project directory: `cd CS510_Project`
- Optional: Create and activate a virtual environment:
  - Create the virtual environment: `python3 -m venv env`
  - Activate the virtual environment: `source env/bin/activate`
- Install backend dependencies using the setup script: `./setup.sh`
- Configure the application by creating and modifying the `app.yaml` file according to the project requirements. ("A full example can be found in the README.md)
- Start the backend server: `python3 api.py`
- Load the extension in Chrome:
  - Navigate to `chrome://extensions/`
  - Enable Developer Mode
  - Click on "Load unpacked" and select the "extension" folder from the project directory

8.2 Use Scenarios

- Open the YouTube website and click on any video.
- Use the site as normal. Note that more interactions with the site such as scrolling through comments, liking an element etc will provide the best results.
- Click on the extension icon to open the popup and view tailored recommendations based on the content and interactions.
- Explore the recommendations by clicking on keywords or videos. Keywords will open a new YouTube tab and search for that keyword while clicking on the video will open that content directly.

REFERENCES

[1] 2024. KeyBERT. https://maartengr.github.io/KeyBERT/ Accessed: 2024-05-09.

[2] 2024. ml6team/keyphrase-extraction-distilbert-inspec. https://huggingface.co/ml6team/keyphrase-extraction-distilbert-inspec Accessed: 2024-05-09.

[3] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. YAKE! Keyword extraction from single documents using multiple local features. *Information Sciences* 509 (2020), 257–289. https://doi.org/10.1016/j.ins.2019.09.013

[4] Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*. Morristown, NJ, USA.

[5] Megha Kulkarni, Debanjan Mahata, Rajiv Arora, and Rajdeep Bhowmik. 2022. Learning rich representation of keyphrases from text. In *Findings of the Association for Computational Linguistics: NAACL 2022*. Stroudsburg, PA, USA.

[6] Flori Needle. 2022. TikTok Tests New Search Features: What We Know Right Now. https://blog.hubspot.com/marketing/tiktok-seo Accessed: 2024-05-08.

[7] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extraction from Individual Documents*. 1 – 20. https://doi.org/10.1002/9780470689646.ch1

9 CONTRIBUTION

- Zongxian Feng: Develops and trains the NLP model for keyword extraction.
- Chenxin Wang: Handles frontend development of the browser extension.
- Andres Suazo: Manages backend integration and connects the NLP model with the frontend.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009