# NLP Final Report

**Authors:** Carlos Figueroa-Díaz, Cristian León Meza

**Date:** May 2025

**Github Repo:** https://github.com/caf3676/NLP-Project

**NLP Final Report: Knowledge on Demand: Query Retrieval and Analysis using LLMs**

# Overall Goal

Our project aims to help students, particularly visual learners, efficiently find high-quality YouTube videos that explain complex topics. Instead of manually searching through numerous videos, the system automates the process by evaluating a ranked list of relevant videos based on factors such as textual similarity, keyword relevance, readability, and engagement metrics. This ensures that learners receive the most useful content without the frustration of ineffective searches.

# Scope

For this class, we developed a functional prototype that processes user queries, retrieves relevant YouTube videos via the YouTube Data API, and converts speech to text using Whisper. It ranks an extracted set of YouTube videos using a multi-factor ranking system to generate and display a ranked list of videos with relevance scores. These metrics are text similarity, keyword matching and ranking, text complexity, and video engagement. Each metric utilizes various machine learning and NLP libraries. These include BERT transformers for text similarity, BM25 for keyword matching, textstat for text complexity, and spaCy for query preprocessing. We were successful in scaling the prototype to handle any search query assuming there is a relevant pool of YouTube videos on the topic. Before the query is evaluated against the video transcripts, it is filtered for irrelevant keywords using parts of speech tagging in spaCy. Upon calculating the quality scores for each video, the model returns the video link and title with the highest score and uses a LLM model (Groq) to provide a summary of the video. The entire project uses a Streamlit user interface accessible through a webpage for visualization.

# Contributions

**Carlos Figueroa-Díaz** Carlos was responsible for integrating the YouTube Data API into the model to obtain the YouTube videos used for ranking. He created Python functions that would extract a set of YouTube links from a query, process those links into audio files stored in a temporary directory using the PyTube library, and transcribe the audio into text files with Whisper. Functions to collect the transcript into an array for further evaluation by the quality metrics, score the text similarity of each transcript using BERT word embeddings and cosine similarity, and tokenize the text to be keyword scored with BM25 were also implemented by Carlos. With the rest of the quality metrics, LLM summary, and Streamlit integration handled by Cristian, Carlos further contributed to the project by writing functions to extract relevant data from the videos such as views, comment count, and video quality, the quality score equation, and the final function that returns the most relevant YouTube title and link.

**Cristian León Meza** Cristian was critical towards ensuring that the model not only functioned properly but also had a visually appealing user interface that was easy to use. To accomplish this, he came up with the idea to integrate the entire program with the Streamlit library, allowing the entire program when running to display a webpage with a search bar that allows users to enter queries, see the YouTube title and link, along with a summary of the video. This summary is provided by the Groq LLM which analyzes the transcript of the recommended video. Beyond these end stage tasks, Cristian also played a vital role in implementing the text complexity and engagement scoring metrics, which required careful use of the spaCy libraries and the creation of custom formulas for generating the engagement score. Cristian also modified the model to filter extraneous information from search queries and did the majority of the testing of the finished prototype.

# Detected Issues

Many of the issues we detected in this project were due to the nature of the YouTube Data API. Sometimes the videos the model would attempt to extract were livestreams for shortform content like YouTube Shorts, which resulted in incomprehensible transcripts for the former, and a lack of text to analyze for the latter. We suspect this occurred because YouTube Shorts tend to have higher view counts than normal videos and trend higher on the search algorithm YouTube Data API uses, and livestream titles sometimes contain many relevant keywords shared with the query. The Whisper model we used also did not always provide the most accurate transcription of the collected audio, resulting in a less accurate quality score for certain videos. Although we could use a more advanced model, it would reduce the speed of the prototype, which already was slower than anticipated due to hardware limitations. Finally, for certain queries, the model had trouble obtaining videos that correlated with the topic at hand, which diluted the pool of potential high-quality videos that could be returned to the user. We surmise that improved query preprocessing could mitigate this issue.

# Lessons Learned

**Carlos Figueroa-Díaz** Through this project I had the opportunity to familiarize myself with many machine learning and NLP libraries that I have never utilized in a project. Learning how to convert strings into word embeddings using BERT was an exciting application of concepts I learned in class. This was also the first time I implemented an API in a project, which proved to be challenging as I had to read documentation and judiciously use the limited pool of resources I had when testing and debugging. As far as lessons learned from the challenges we faced in this project, I would say the most valuable lesson I learned is how sensitive an NLP model is to certain keywords and how the volume of information fed into a model does not necessarily correlate with its efficacy. A potential improvement for this project would be integrating a memory bank for the LLM the model uses so that it can draw from previous queries for better preprocessing in the future.

**Cristian León Meza** Compared to other projects I have worked on in the past, this prototype was challenging in the sense that it was an amalgamation of almost every concept we have learned in this class via a pipelined approach. I learned that you cannot simply throw together a collection of libraries and expect a model to work as expected, rather it takes careful analysis of inputs and outputs until an end result is achieved. The biggest challenge I faced was figuring out how to preprocess the queries so that irrelevant information would be removed without eliminating keywords useful to the model. This required going back through what we learned about lemmatization, parts of speech tagging, and other text extraction techniques covered earlier in the semester, which reinforced my learning. Aside from NLP, I also learned that developing a clean user interface requires a lot of debugging to iron out the details. I believe we could improve this prototype by modifying the summaries provided by the LLM so that they explain why the top video was chosen, along with suggestions about the other ranked videos based on what criteria they scored high on relative to the other videos.

# Self-Scoring Table

**Carlos Figueroa-Diaz**

|  |  |
| --- | --- |
| **Grading Category** | **Explanation** |
| 80 points – Significant Exploration Beyond Baseline | Detected issues related to video metadata extraction and fixed API related bugs. Also tested queries using models of increasing quality to achieve better results. |
| 30 points – Innovation or Creativity | Came up with custom scoring metrics for recommending the highest quality YouTube video on a given topic. These metrics were tweaked based on observations gathered from various research papers on similar models. |
| 10 points – Highlighted Complexity | Performed text similarity and keyword matching scoring using transformers and bag of words retrieval on videos collected from YouTube data API. Also came up with a way to extract video transcripts from audio collected from YouTube links without clogging file space. |
| 10 points – Discussion of Lessons Learned and Potential Improvements | See discussion section above. |
| 10 points – Exceptional Visualization | Project contains a clean user interface via Streamlit. |

**Cristian León Meza**

|  |  |
| --- | --- |
| **Grading Category** | **Explanation** |
| 80 points – Significant Exploration Beyond Baseline | Discovered the issue of livestreams and YouTube shorts appearing. Performed extensive testing of queries and modified query masking for better prompting. |
| 30 points – Innovation or Creativity | Modified the model to provide an LLM based summary alongside the recommended video. Noticed that users sometimes preferred the brevity of a summary rather than the full video depending on the circumstances. |
| 10 points – Highlighted Complexity | Obtained data from testing queries to help modify query masking and improve model parameters for quality scoring. This added complexity allowed for improvements on user experience (UX). |
| 10 points – Discussion of Lessons Learned and Potential Improvements | See discussion section above. |
| 10 points – Exceptional Visualization | Project contains a clean user interface via Streamlit. |

# Data Sources

Krishna Choudhari, Vinod K. Bhalla, Video Search Engine Optimization Using Keyword and Feature Analysis, Procedia Computer Science, Volume 58, 2015, Pages 691-697.

Nock, H.J., Iyengar, G., Neti, C. (2003) Issues in Speech-Based Retrieval of Video. Proc.

ISCA Workshop on Multilingual Spoken Document Retrieval (MSDR 2003), 67-72.