Software Engineer - ML and Search - Coding Exercise

Vimeo has hundreds of millions of videos on our platform. Our curation team hand selects "Staff Picks" and once a month the whole company watches and votes on a staff pick of the month which is featured on the site.

A product manager wants to create a system to suggest "similar" staff pick videos when you complete the current staff-pick. Your mission, should you choose to accept it is to write a program that takes a clip_id (aka video_id) as input and outputs "similar" clips based on the clip's title, description, and/or category. We've included metadata of around 4,000 staff picks in a dataset along with their categories they belong to and links to their thumbnails which you can download here: https://drive.google.com/open?id=1zbCF5jVmc2Gy5L1_JqJICGvbhRexwWaX

To complete the exercise, your program must:

- Accept a clip_id as the only argument on the command line
- Return a list of the 10 most similar videos in the dataset in order of similarity given a single clip_id (also from the dataset.)
- The format of the results returned should be an ordered JSON list of JSON objects with the following fields for each clip result:
    - id
    - title
    - description
    - categories (comma separated list)
    - image (url of thumbnail)
- Please add a single JSON file called `results.json` to your submission showing the results for the following clip_ids in your submission where the keys are the following clip_ids:
    - 14434107, 249393804, 71964690, 78106175, 228236677, 11374425, 93951774, 35616659, 112360862, 116368488

Suggested Approaches:

- Use an inverse index and/or TF-IDF information directly find closest matches.
- Generate a vector/embedding and find closest vector matches. For instance:
    - Use an existing word embedding model like Word2Vec to create vectors.
    - Or vectorize the TF-IDF information and use that as your vector.

Also include a single-page writeup discussing:

- Your approach used and what you liked and disliked about this challenge.
- How would you go about building this for real at Vimeo?
- Are there additional techniques you would leverage to improve the results?
- What performance considerations are important to scale this up to all Vimeo videos and why?

Things we care about for this project:

- You create something that at least works in the time allowed.
- Clean, readable code
- You have fun while building this.

Bonus points for:

- Implementing an additional web interface where you can submit any clip_id in the dataset and get the similar clips JSON as a response.
- Submitting your work as a git repo with clean commit messages.
- Documenting your code
- Writing Tests

Datasets

similar-staff-picks-challenge-clips.csv

Description: around 4000

id	clip ID
title	Clip title
caption	Clip description
created	Created date
filesize	Original video file size in bytes
duration	Clip duration in seconds
clip_id	Clip ID
total_comments	Total number of comments on this video

total_plays	Total number of plays on this video
total_likes	Total number of likes on this video
thumbnail	http address of the video thumbnail

similar-staff-picks-challenge-clip-categories.csv

Description: Maps clips to the categories it belongs to.

clip_id	ID of the clip
categories	comma separated list of category IDs

similar-staff-picks-challenge-categories.csv

Description: Vimeo category info.

category_id	ID for this category
parent_id	ID of the "parent" category (the 14 major categories have parent_id=0). Any video has to have at least one parent category before having a sub-category
name	Text name for the category