Content Tagging

Raviteja Anantha Ramesh(rar555)
Ganapathy Aiyer(gsa277)
Karthik Tiruveedhi(vst216)
Kashish Dua(kd1651)

Introduction

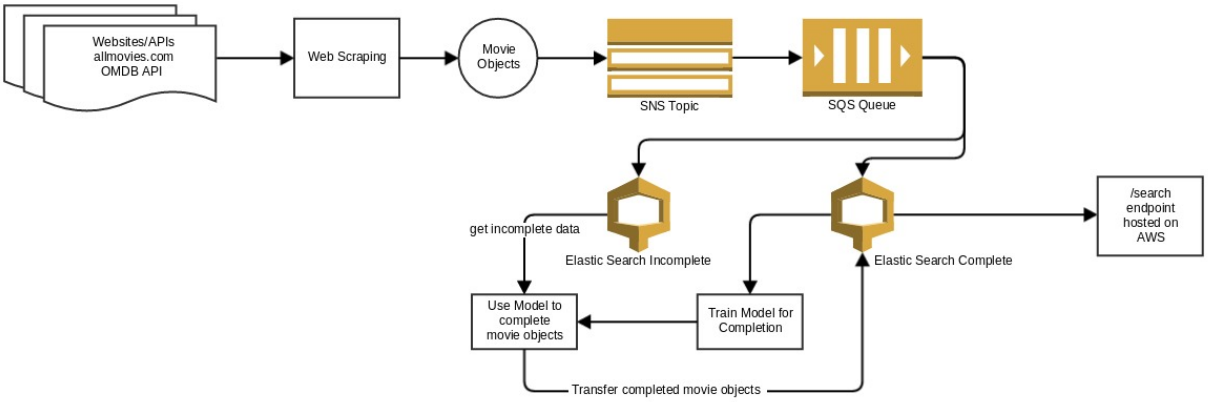
Our objective is to tag the scenes and movies according to their genre. We use data from *allmovie.com*, we scrape the website and get all the movies. For every movie we have Themes, Moods and Keywords associated with the respected movie. We scrape recent movies from 2016 onwards. We use OMDB API in parallel as each movie is scraped to get Genres, Directors and Actors. Note that we may or may not get Keywords and/or Genres. In total, we got 7245 movies in which we have 2856 complete movies and 4389 incomplete movies. By incomplete movies, we mean that either Keywords or Genre is missing. We describe the architecture in the following section and later we explain how we fill the missing data, *i.e.* Keywords and/or Genre in Phase-1 and how we tag each scene in Phase-2, along with accuracy.

Architecture

We scrape the website and use OMDB API in parallel to get both complete and incomplete movies and published to SNS topic. We use SQS worker which periodically consumes movies and uploads to Elastic Search to two different indices based on completeness or incompleteness. Then we use the complete data to train our algorithm and then complete the incomplete data and upload it to complete data list and delete the incomplete data from Elastic Search.

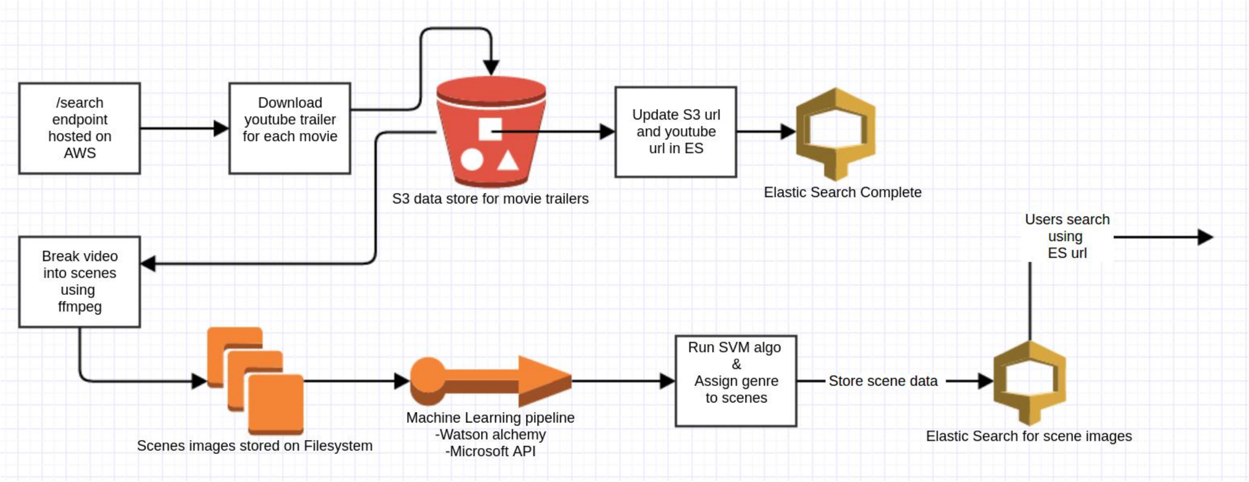
Phase-1

For the incomplete movies we use Stochastic Gradient Descent to fill the information that is missing, such as Keywords and Genres. We split complete movies into two sections *viz.* Training Data, which is 2/3 of the complete movies, and Testing Data which is 1/3 of complete movies. We train Stochastic Gradient Descent on Training Data using Synopsis of the movies and test our accuracy on Testing Data, we got **accuracy of 88.81%**, there were movies that didn't have Synopsis, we ignored those as we can't build on it since we took synopsis as our basis to build our model. We then use our trained model to tag the incomplete movies and fill the missing Keywords.


Websites/APIs
allmovies.com
OMDB API
Web Scraping
Movie Objects
SNS Topic
SQS Queue
get incomplete data
Elastic Search Incomplete
Elastic Search Complete
/search endpoint hosted on AWS
Use Model to complete movie objects
Train Model for Completion
Transfer completed movie objects

Phase-2

Now as we have all the movies in Elastic Search, we query the database and get the names of the movies and programmatically generate Youtube urls for the movie trailers and programmatically download the videos and upload all the videos to S3 along with the S3 link. We get images of the video at certain intervals and pass it to the pipeline we created which uses two IBM Watson APIs and one Microsoft API. We use one IBM Watson to get the entities in the image, and the other IBM Watson API to know whether a human face exists in the image. If we have a human face in the image, then we use Microsoft API to get the emotion. We store all the JSON objects returned by each API in a dictionary. We tried to use unsupervised learning, like using clustering algorithms, but the problem is the feature space varies with huge variance for same tags, and the probability of tagging an image correctly based on similarity/distance measure of feature space is very less. Considering these facts, we decided to approach the problem using supervised learning techniques. We then use Support Vector Machine with Radial Basis Function as its Kernel Function to create a model that tags each image. We created labeled data to train our model. Note that not every each can be tagged, meaning some images might come under no Genre and we tag them with 'No Genre'. Since we don't have labeled data and we created labeled data for training purposes, it's not exhaustive and limits the generalized classifying ability of the model to the data it has been trained on, we predict that the model can perform in a superior way if we have labeled data. Once the tagging is done, we upload the images along with the tags to Elastic Search.


/search endpoint hosted on AWS
Download youtube trailer for each movie
S3 data store for movie trailers
Update S3 url and youtube url in ES
Elastic Search Complete
Users search using ES url
Break video into scenes using ffmpeg
Scenes images stored on Filesystem
Machine Learning pipeline
-Watson alchemy
-Microsoft API
Run SVM algo
&
Assign genre to scenes
Store scene data
Elastic Search for scene images

Future Work:

For the current implementation of scene tagging image that has been taken periodically, we can improve it by taking a bunch of images instead and tagging each among them, and then deciding the tag for the whole scene. One more important enhancement is to use a Deep Convolutional Neural Network instead of Support Vector Machine.

GitHub Link: *https://github.com/RavitejaAnantha/CloudComputing*