# Whatcha Watching?

The objective of this project is to build a advanced information retrieval system to support multiple search methods for people who are interested in TED Talks. The TED Talks is popular by its highly reputed speakers and their inspiring speeches. It is not only a shortcut for people who are interested in learning how to deliver a speech but also an educational material which encourage people to think out of the box. It has high impacts on people from all over the world, and many TED Talks are translated into multiple languages.

However, we notice that *TED* does not support content based search and this may make people who are interested in a specific talk struggling. This project uses TED Talks to build a high performance information retrieval system with multiple features to search, such as free text search which supports title, transcript, and description with doubled weight on title, search by speakers, and search by talks' length.

This system supports a tf-idf relevance based recommendation function which shows users the similar TED Talks on the result pages. The main relevance calculation for results' ranking is based on elasticsearch's default method of BM25.

## Requirements

To implement all the functions of the system, you should have the following:

**General Dependency:**

- Python 3.5+ (required) : used to run the code
- Elasticsearch 6.7 (required) : used to implement the search engine
- Google cloud account (required) : used to scrape data from youtube
- Youtube data V3 API : used to scrape data from youtube
- Word cloud API : used to implement the word relevance map
- Bootstrap : used to implement the web

**Python Libraries Dependency:**

ElasticSearch

```
pip install elasticsearch
```

Flask

```
pip install -U Flask
```

Elasticsearch-dsl

```
pip install elasticsearch-dsl
```

Wordcloud

```
pip install wordcloud
```
Googleapiclient
```
pip install google-api-python-client
```
Google-auth-oauthlib
```
pip install google-auth google-auth-oauthlib google-auth-httplib2
```
Beautifulsoup4
```
pip install beautifulsoup4
```
Requests
```
pip install requests
```
Scikit-learn
```
pip install scikit-learn
```
Json

Re

The following tools are optional, but can improve the efficiency of work:

- Git

## General Layout

The code is divided into three parts, scraper, search engine and frontend.

### 1. Scraper:

TedTalks_scrape.py: scrape the data from TED

YouTube_scrape.py: scrape the data from YouTube and merge it with the data from TED to build the full corpus

The following data are scraped for building this information retrieval system:

"Title": the title of this TED Talk

"Description": the short description of this TED Talk

"Date": the date of this TED Talk being published

"Thumbnails": a image of the speaker of this TED Talk

"Tags": a set of descriptive tags of this TED Talk

"Num_views": number of views of this TED Talk from *TED* and *YouTube*

"Num_comments": number of comments of this TED Talk from *TED* and *YouTube*

"num_transcripts": number of languages available for transcripts/subtitles in this TED Talk

“Categories”: feedback of users on *TED*

“Transcript”: English transcript of this TED Talk

"YouTube_likeCount": number of likes from *YouTube*
"YouTube_dislikeCount": number of dislikes from *YouTube*
"YouTube_favoriteCount": number of favorite from *YouTube*
"Comments": comments from *YouTube*

## 2. Search Engine:

/ ted_index.py
/ ted_query.py
Parsing the corpus according to the fields we need and build index of the corpus. Using elasticsearch to do the query and return the result, containing parameters of the page, to frontend.

According to the different weight of the fields, allocate different weight to the fields. Highlighting keywords of query and return parts of the results.

## 3. Frontend:

/templates/query.html
/templates/result.html
/templates/talk_page.html
/static/css/query.css
/static/css/result.css
/static/js/bootstrap.min.js
/static/js/circle-progress.min.js
/static/js/jquery-3.2.1.min.js
/static/js/main.js
/static/js/map.js
/static/js/owl.carousel.min.js
/static/img

Three html pages act as entrance of search, showing the structured and filtered data about the talks. List of the result is shown in the result page and users can see the details of each video in specific page corresponding to each talk.

Other css/js files used to build the pages.

## 4. Readme.md

Instruction of the program.

get_authenticated_service(): used for accessing to the API
search_by_channelId(service, **kwargs): run API to query over YouTube by TED Talks channel id
write_video_id_list_to_file(outfile, service, **kwargs): call API to write the YouTube video id defined by YouTube into the outfile separated by \n
write_videos_statistics_to_file(id_list, outfile, service, **kwargs): call API to write the videos statistics (viewCount, likeCount, dislikeCount, favoriteCount, commentCount) to file
get_video_comments() call API to get comments based on YouTube video Id

## Design

1. Scrape:

- With the purpose of showing the best search results for users who are interested in TED Talks, we use both data from *TED* and *YouTube* to build our corpus to get a more comprehensive perspective of each TED Talk.
- *TED* has a total number of 3842 TED Talks while *YouTube* has only 2973 TED Talks.
- *YouTube* is much more popular than *TED* worldwidely, *YouTube* has more views and more comments with features of likes and dislikes.
- Through observation, almost all TEDers' comments on TED Talks are positive. The reason why this happens might be that only people who love watch TED Talks go to *TED.com*. In contrast, youtubers do not mind to leave negative comments and criticize the speakers' ideas. Therefore, we only take comments from *YouTube*.
- Originally, we also get the number of likes for the comments. However, most comments with high number of likes are not the review of the content of TED Talks, instead, general they make jokes and have a good sense of humor. Therefore, we use the relevance between the comments and the TED Talks to rank the comments.

2. Search Engine:

- We use about eighteen fields to help us make an overall search about ted talk from many aspects, like score, speaker, number of views, number of comments, posted date and ratings.
- Firstly we directly show the fields about ted talks that related to the query words. Each video in the result list has its own fields, like title, speaker, views, posted date etc. and parts of the description of the video and the link of source page are also displayed in the result page.
- Also, we use some raw data to do some manipulation and then analyze whether the elements we get is related or typical to the video. For example, when we do some preprocessing. we use views, scores and the number of comments to calculate the relevance and popularity as additional elements of the videos we need to take into consideration.
- We build a recommendation system according to the titles, topics, description and transcript using tf-idf model. Initially, we tried to allocate different weights to them and compare the results, and the best choice is to connect them as a string before calculating the score.
- Extract ten related talks and transfer the title, pic, link that connected to the talk_page to frontend.
- Word cloud by using the feedback from users of TED to count the audiences reaction to the TED Talks.

3. Frontend:

- Users can search ted talks by different fields, free text search is necessary. They can also search the talks by specific speaker. Another constraint is duration, which means how long the talk is.
- In the result page, we need to show the information related to each video. Basically, title, speaker, views and other information like this are shown just besides the talk, and users can directly go to the source page of the ted talk or click the title to see the details of the talk.
- In the talk page, details of the talk are shown, such as the word cloud, transcript and recommended talks. Users can also watch the video in this page. Related talks are list in this page and user can enter the talk_page by clicking the pic of each video.

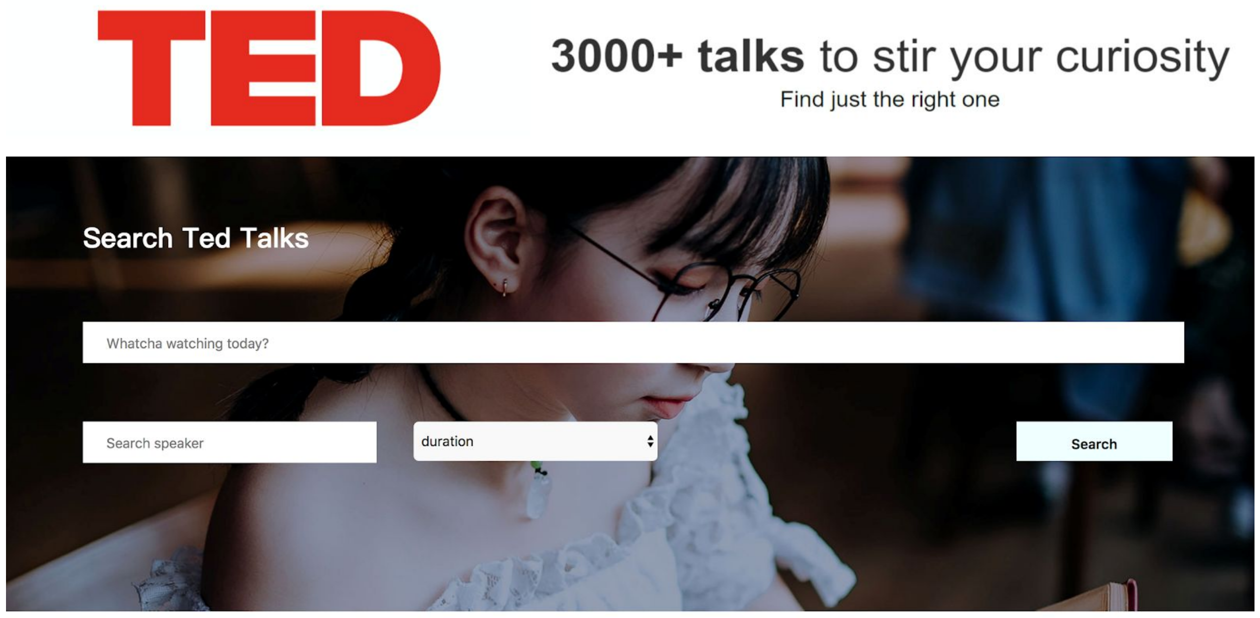
## Compiling and Running Program

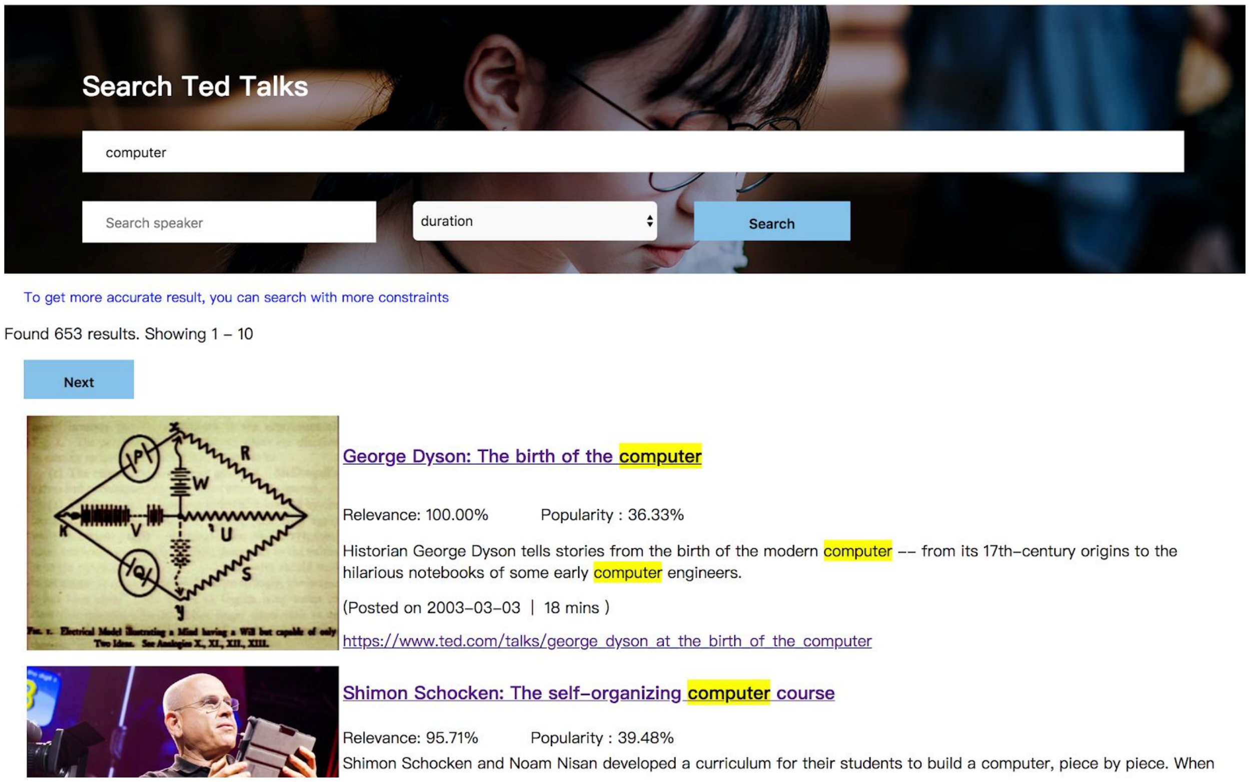
- Start elasticsearch service before building corpus.
- Create corpus needed as structured data, run ted_index.py to build structured corpus.

- Run ted_query.py, and open the search page http://localhost:5000/results, shown in screenshot below.
- Search by free text, speaker and duration. Tests are as follows.

## Test

1. Using free text to search.




Search Ted Talks

computer

Search speaker

duration

Search

To get more accurate result, you can search with more constraints

Found 653 results. Showing 1 – 10

Next

George Dyson: The birth of the computer

Relevance: 100.00% Popularity : 36.33%

Historian George Dyson tells stories from the birth of the modern computer –– from its 17th–century origins to the hilarious notebooks of some early computer engineers.

(Posted on 2003–03–03 | 18 mins )

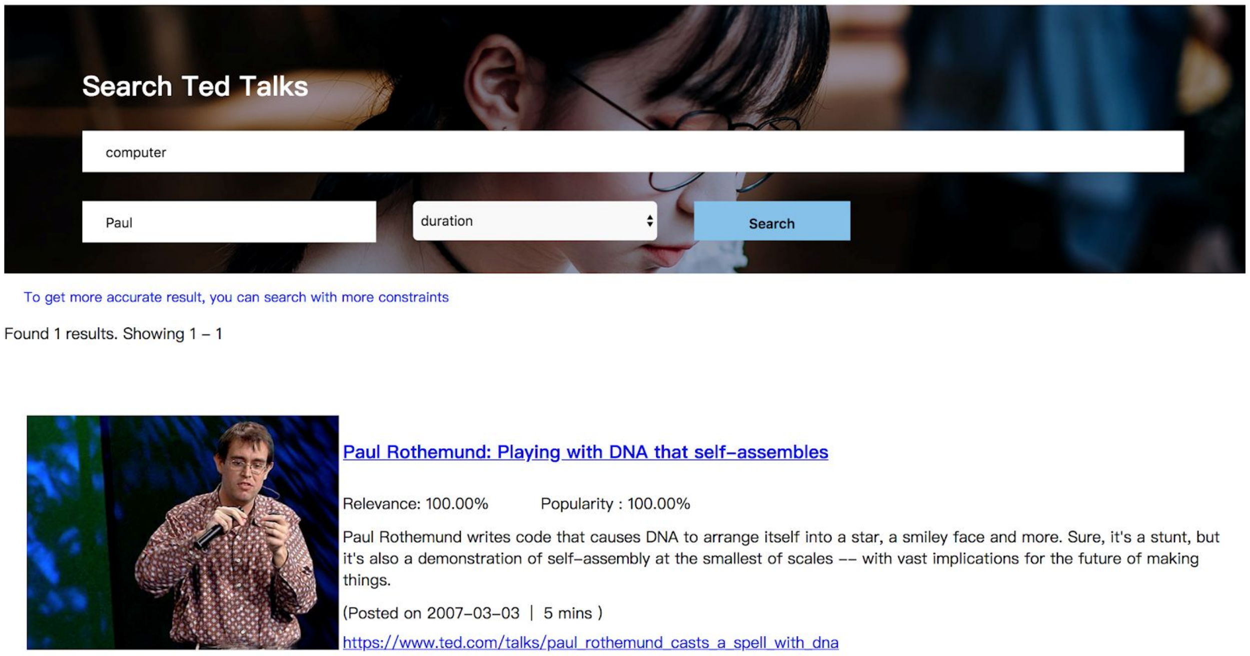
https://www.ted.com/talks/george_dyson_at_the_birth_of_the_computer

Shimon Schocken: The self–organizing computer course

Relevance: 95.71% Popularity : 39.48%

Shimon Schocken and Noam Nisan developed a curriculum for their students to build a computer, piece by piece. When

2. Using speaker to search.


Search Ted Talks

computer

Paul

duration

Search

To get more accurate result, you can search with more constraints

Found 1 results. Showing 1 – 1

Paul Rothemund: Playing with DNA that self–assembles

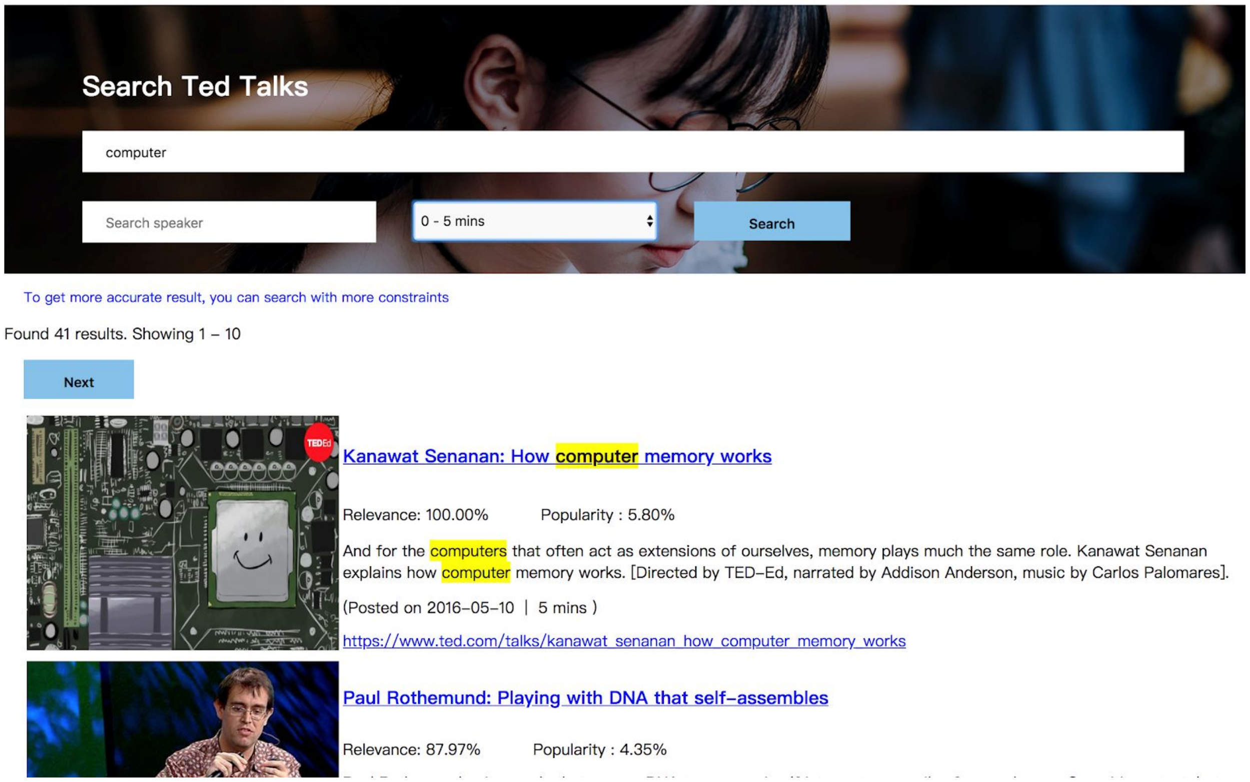
Relevance: 100.00% Popularity : 100.00%

Paul Rothemund writes code that causes DNA to arrange itself into a star, a smiley face and more. Sure, it's a stunt, but it's also a demonstration of self–assembly at the smallest of scales –– with vast implications for the future of making things.

(Posted on 2007–03–03 | 5 mins )

https://www.ted.com/talks/paul_rothemund_casts_a_spell_with_dna

3. Searching with duration.



Search Ted Talks

computer

Search speaker

0 - 5 mins

Search

To get more accurate result, you can search with more constraints

Found 41 results. Showing 1 – 10

Next

[Kanawat Senanan: How computer memory works](https://www.ted.com/talks/kanawat_senanan_how_computer_memory_works)

Relevance: 100.00%     Popularity : 5.80%

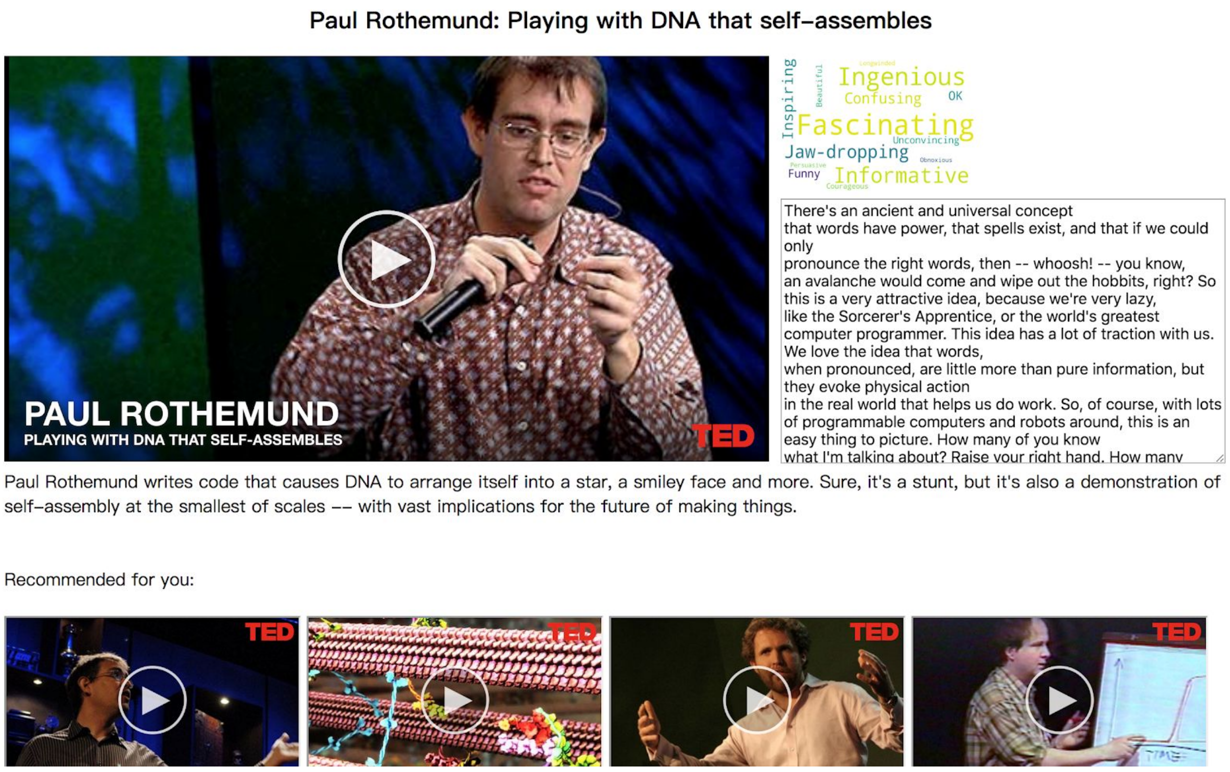
And for the computers that often act as extensions of ourselves, memory plays much the same role. Kanawat Senanan explains how computer memory works. [Directed by TED–Ed, narrated by Addison Anderson, music by Carlos Palomares].

(Posted on 2016–05–10 | 5 mins )

https://www.ted.com/talks/kanawat_senanan_how_computer_memory_works

Paul Rothemund: Playing with DNA that self–assembles

Relevance: 87.97%     Popularity : 4.35%



Paul Rothemund: Playing with DNA that self–assembles

There's an ancient and universal concept
that words have power, that spells exist, and that if we could only
pronounce the right words, then -- whoosh! -- you know,
an avalanche would come and wipe out the hobbits, right? So this is a very attractive idea, because we're very lazy,
like the Sorcerer's Apprentice, or the world's greatest computer programmer. This idea has a lot of traction with us.
We love the idea that words,
when pronounced, are little more than pure information, but they evoke physical action
in the real world that helps us do work. So, of course, with lots of programmable computers and robots around, this is an easy thing to picture. How many of you know
what I'm talking about? Raise your right hand. How many

Paul Rothemund writes code that causes DNA to arrange itself into a star, a smiley face and more. Sure, it's a stunt, but it's also a demonstration of self–assembly at the smallest of scales –– with vast implications for the future of making things.

Recommended for you:

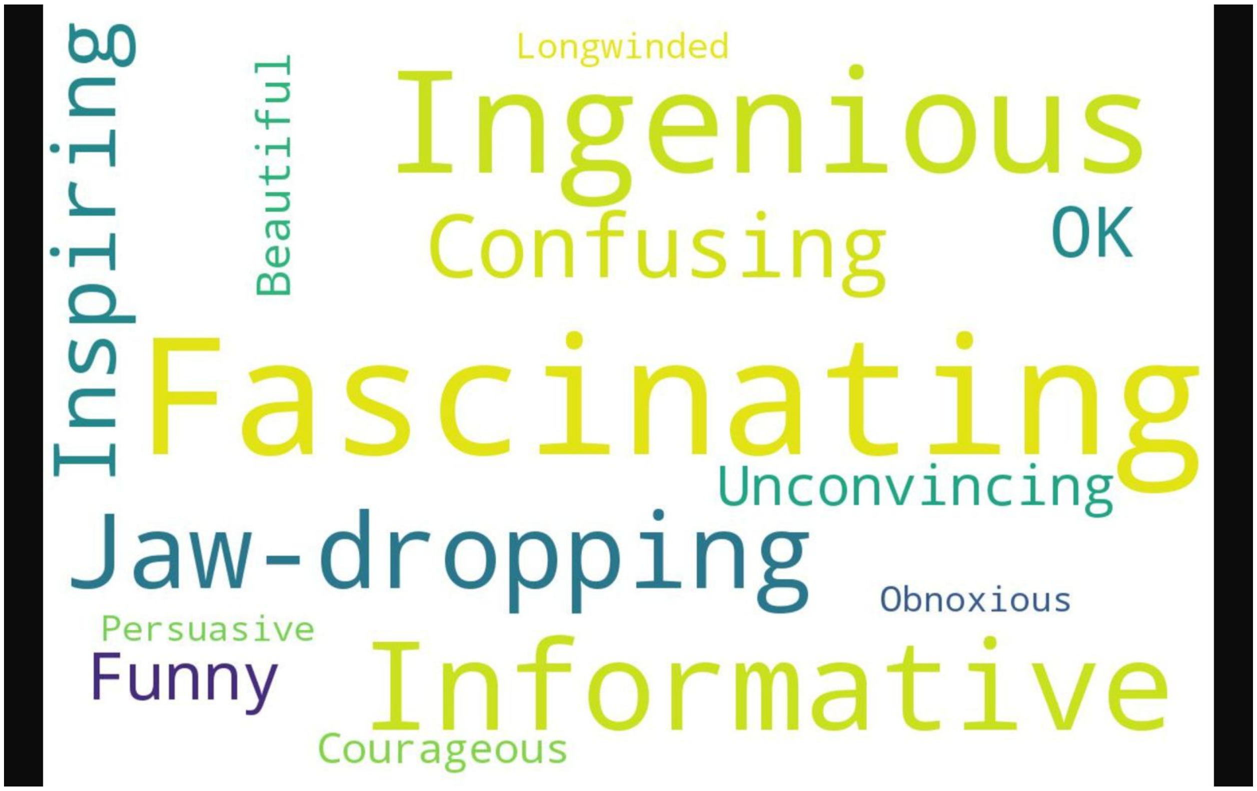
## Omar Ahmad: Political change with pen and paper

Obnoxious
Persuasive
Inspiring
OK
Ingenious
Informative
Jaw-dropping
Funny
Unconvincing
Fascinating
Courageous

One of the things that defines a TEDster is you've taken your passion, and you've turned it into stewardship. You actually put action to the issues you care about. But what you're going to find eventually is you may need to actually get elected officials to help you out. So, how do you do that? One of the things I should probably tell you is, I worked for the Discovery Channel early in my career, and that sort of warped my framework. So, when you start to think about politicians, you've got to realize these are strange creatures. Other than the fact that they can't tell directions, and they have very strange breeding habits, how do you actually work with these things? (Laughter) What we need to understand is: What drives the political creature? And there are two things that are primary in a politician's heart: One is reputation and influence. These are the primary tools by which a politician can do his

Want your local politician to pay attention to an issue you care about? Send a monthly handwritten letter, says former mayor Omar Ahmad -- it's more effective than email, phone, or even writing a check. He shares four steps to writing a letter that works.

What people say on Youtube:
Great presentation. Keep it coming. Thanks. -- David Sunn
Mr. Ahmad, our Mayor, passed away today. It was like a sucker punch. It hurts because it was so unexpected and he was so young. He was an inspirational citizen and worked diligently and intelligently for our community. It definately hurts that he is no longer here. This is the first time I've seen this talk. He was like this, humorous but with a purpose in his communication and a problem solver. I'm sorry for his family's loss and our community's loss. -- Rhon
this is how we catch the bull by the horns btw bull = politician : D -- ahmeds027



## Challenge

During the development, our team faces a series of challenges of building a high performance information retrieval system. The first challenge is evaluation. Our information retrieval system is built offline and it is extremely difficult to find a good way to evaluate the performance of our system. As a team with only three members, it is nearly impossible to perform a high quality test between different system settings. The state of the art information retrieval system like Google uses A/B testing which can get more compelling results. The second challenge is to develop a beautiful UI. None of our team members is expertised in UI design. The third one is that the time is limited and our teams cannot get many feedbacks from peers. Besides, Google Cloud Youtube Data V3 API has a lot of constraints to prevent developers to get the data. For example, one can only query over YouTube through programming languages 100 times per day. Also, the search results have the maximum number of results of 500 which is the way below our required corpus size.

## Future Work

We have come up with three ways to further improve our information retrieval system.
The first one is to do a query correction which acts like what Google does. However, we do not have a large database and server to record the users habits.
The second way is to apply natural language process (NLP) knowledge of sentiment analysis to rank the comments from youtubers and get the most representative two with one expresses a positive review and another expresses a negative review. However, it is not easy since this task need to classify not only sentiment but also the quality of the reviews.

## Contributors

- Kuan-Yu Chen(Leader) <chenky@brandeis.edu>
- Zhiheng Wang <zhihengwang@brandeis.edu >
- Cheng Chen <xiaoxiaochen@brandeis.edu>