POLITECNICO DI TORINO

Master's Degree in Automotive Engineering

Master's Degree Thesis

Book annotation with videos using Cross-media retrieval

Supervisors

Prof. Laura FARINETTI

Dott. Lorenzo CANALE

Candidate

Shao Bing

July 2021

Summary

As the rapid progress of the e-learning platform and the bloom of on-line educational resources, there are more and more needs from different aspects to improve the quality and feeling of the study in this age of Internet and multi-media. In this context, this thesis proposes and creates an approach to enable the annotation of the traditional books in PDF format with abundant educational videos on YouTube.

Here the meaning of the word annotation is that when a student or reader has some doubts on some keywords or concept in a PDF book, then the relative chapter title has already been linked to a list of relevant YouTube videos which may help to understand the doubted keywords comprehensively.

As a result, in order to achieve the target descripted above, the techniques from a field of computer science called cross-media retrieval should be considered and utilized. More in details, the field cross-media retrieval actually belongs to a bigger field called information retrieval, which can be briefly summarized as the process of documents retrieval according to a specific query. And one of the most famous instances of the implementation of information retrieval is the web search engine, such as google search engine or baidu search engine.

Thus, the thesis first investigates and compare different types of methods that do cross-media in different ways, such as deep learning methods, learning to rank methods and graph-based methods.

There are many tackles along the way to achieve the mentioned goal, first, There isn't any pipeline or system that link the text in PDF books and videos from an online platform before, thus it should first be done by ourselves, which means we should first recognize the chapter titles (keywords or important sentences) in PDF format books, and then retrieve video results from YouTube using the extracted chapter titles as queries, and finally improve and select only best ones to be linked as the final results.

Thus, along the journey of this thesis, many different kinds of tools and methods have been utilized. First, in order to extract text contents from PDF books, a python package pyPDF has been used. Then to limit the range of the search or do a customized search on YouTube, google custom search engine has been used. And to extract subtitles of videos, pyTube has been used. After all the previous preparation steps, we do the feature engineering work and are train a xgboost model to classify whether a video result is related or not automatically.

In the end the cross-media retrieval has been done through all the steps mentioned above, and the quality of ranking lists of the prepared queries from PDF books has been globally improved by around 10%.

Acknowledgements

Here I want to express my gratefulness to my parents who always support me all the time. And I want to express my gratefulness for my friends both in China and Italy, who always give me any kind of help when I need. And for the work of this thesis, I'm really grateful to Professor Laura who gave me the chance to do some research in the field that I'm interested in. Moreover, I'm especially very thankful to my college and friend Lorenzo, since he did every-thing he could do to make this work well-being during these special days of Covid-19. And from him I can also learn to be smarter in the work and more open-mind in the life. And as a foreign student, I will never resist to express my likeness and love to Politecnico and Italy.

Shao Bing

Table of contents

Chapter 1

Introduction

In the recent years, educational resources become more and more multi-modal as the world of the Internet blooms. And the students from different phases of study also become more and more familiar of the use of multi-media learning material. Especially in these days, the current COVID-19 pandemic dramatically impacts the practice of education [1], [2], which forces on-line courses to be necessary almost anywhere. And this impact will bring both challenges and opportunities for a long period [3].

More in details, most of the universities all over the world have been forced to provide remote or a mixed learning modality, which makes teaching sessions streamed and then recorded for the sake of those students that are not able to attend them in campus. This dramatically increases the quantity of available video educational material, which can be really valuable for the future learners [4], [5].

As a result, it is evident enough that the progress of cross-media retrieval methods, capable of retrieving multi-modal resources at the same time can be really helpful in this age of information overloaded. And the recent progress of the fields Natural Language Processing and Deep Learning also enhances the possibility of automatically linking educational resource in different modal having semantics in consistency [6]. On the one hand, techniques of the former field provide capable sematic representations and handle of text. On the other hand, techniques of the latter field make it possible to learn models from large-size dataset which can provide black-box but powerful representations of multi-modal resource [7].

Since the close relationship with semantic web techniques, cross-media retrieval systems can be really helpful for learners to create their own knowledge system of a specific domain of inter-domains [8]. In addition, learning from materials in different modal also makes students much more engaged during the procedure of their study [9].

And the work presented in this thesis mainly addresses the solution of how to facilitate learners that are studying with either text books or online educational videos to retrieve relevant resource of the other modal. One of the major challenges is that educational videos online are usually not annotated with abundant enough semantic metadata, i.e., most of the online video courses do not provide subtitles. And sometimes, even though there is the subtitle, there still isn't any annotation about the key concepts mention in the text of the subtitle. Thus, the need of the use of querying techniques to link multi-modal resource seems to be natural.

And in this work, in order to fulfill the need of an automatic annotation system of text books with videos, the engineering work and experimental parts have been done step by step. We started from collecting data resources, then created text queries and retrieved YouTube video results, next obtained captions of result videos and extracted features from each pair of query and result, finally trained machine learning models to filter out not matched videos and re-calculate a metric called mean average precision to measure whether the quality of result ranking improved or not after filter.

The article is organized as following: Chapter 2 introduces the background of the project more in details，especially some basic introduction of cross-media retrieval. Chapter 3 introduces the methodology proposed and the tools that have been used. Chapter 4 introduces the experiment procedure in details and result analysis. And Chapter 5 draws the conclusions of the thesis and depict the further insights for future work.

Chapter 2

Background: Cross media retrieval

2.1 Cross media retrieval

The work in this thesis is an application of cross-media retrieval techniques manipulating educational books and videos. Hence, we will introduce the field of cross-media retrieval briefly in the following parts of this chapter as the background of the work in the thesis.

As the come of the age of information explosion in this decade, there were at least two major changes occurred. The fist change was that the e-books became more and more available and popular. The other one was that there were more and more educational videos on different platforms, such as MOOC, TED and even YouTube. As a result, it became much more convenient for the experiment and application of cross-media retrieval.

Cross-media retrieval is a task of information retrieval, where the goal is to facilitate users to make queries of a certain modal (e.g., text) and to retrieve relevant results of a different modal (e.g., videos) [6], [10].

The current mainstream method in cross-media retrieval is to learn a common space which is a projection space of multi-modal information resource. Thus, resource of different modal can be projected to the same common space so that they can be contrasted conveniently.

According to [6], there are mainly 7 categories of existing methods of common space learning. Among the 7 categories of the methods, the basic foundation of common space learning is the traditional statistical correlation analysis, which optimizes statistical values to learn linear projection matrices for common space. And the others can be classified as the following:

- Deep learning methods, DNN-based methods take deep neural network as the basic model and aim to make use of its strong abstraction ability to mine complex correlations among cross-media content (e.g., [11], [12], [13], [14]). And a survey on DL-based methods is [15].

- Graph-based methods, which uses weighted graph to model the correlations of multi-modal resource [16], [17]. Depending on the strategy used to extract the resources that are most pertinent to a specific given query, graph-based methods can be further partitioned into graph regulations strategies (e.g., [18], [19]) and neighbor analysis methods (e.g., [20], [21]).

• Learning to Rank methods, which is very different from other methods on the way of training the model. Since it directly optimizes the ranking of the searched results by using the ranking as the training data. Thus, learning to rank methods reformulates the retrieval problem as a ranking problem. (e.g., [22],[23],[24])

• Hashing methods, which takes advantage of generating hash codes for multi-modal resource and projecting cross-media data into a common Hamming space. As the quantity of multi-media data is growing rapidly, which requires high efficiency of retrieval system. Hashing methods are designed for accelerating retrieval process. (e.g., [25], [26])

• Natural Language Processing (NLP)-based methods, which processes textual information extracted from different modalities for the convenience of comparing similarities among resource of different modalities. (e.g., [27], [28])

Here is an example of the generalized architecture of a cross-media retrieval system:

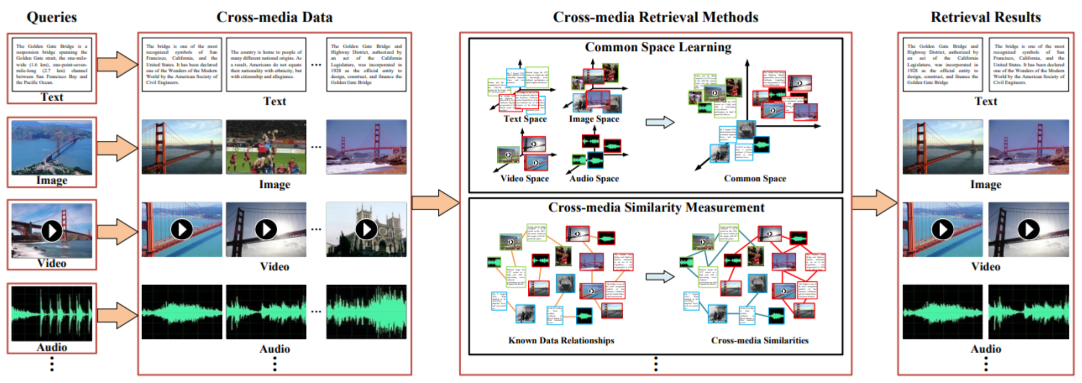


Figure 2.1 an example architecture of generalized cross-media retrieval system.

And in the rest parts of this chapter, we will talk about several examples of each kind of methods mentioned above.

2.2 DNN-based methods

2.2.1 Example 2: Deep correlation for matching images and text

This work is from [12], and is based on a prior state of the art work on text-image matching problem Deep canonical correlation analysis (DCCA) [2.2.2-1].
And here we will briefly introduce this work as another example of DNN-based methods of cross-media retrieval.

The background of this work is the field of describing visual data with natural language, which naturally needs the functionality of cross-media matching. Although generating natural language description for image and video has been a popular research topic in these years, and there are already different methods for the generation of text description, the main issue with description generation is still the lack of automatic and objective evaluation metric.

As a result, the work in [2.2.2-1] was proposed to be this kind of evaluation metric of text description generation for images or videos. More in details,
There is a popular method for the problem of cross-media matching called Canonical correlation analysis, which is able to learn a semantic representation to web images and their associated text. Then the learnt semantic space provides a common representation and enables a comparison between the text and images. Thus, deep CCA is a DNN version of traditional CCA, which means DCCA optimizes the CCA objective in the deep learning framework. It uses the insight that the total correlation sought in CCA can be maximized by optimizing a matrix trace norm.

And the main contribution of this work is that it provides a specific structure to represent separately both the text and image source with deep neural networks as complex features, then both the text features and image features become the input of the final trace norm objective function.

Here is the architecture of the proposed network:

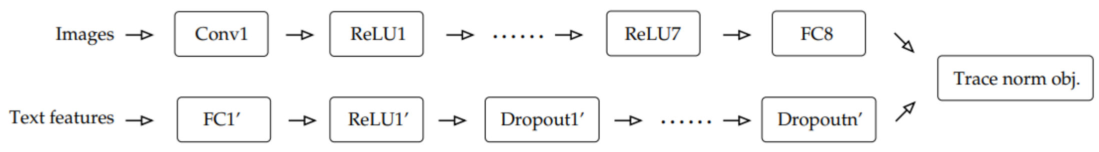


Figure 2.2.2 the architecture of proposed variant DCCA.

From the figure above, we can easily recognize that there are two separate branches in the architecture. The top row in the figure is the image branch, which is essentially the deep convolutional neural network (CNN) proposed in [2.2.2-2]. And the bottom row in the figure is the text branch, which consists of N stacked triplets of fully connected (FC) layer, rectified linear unit (ReLU) layer, and dropout layer.

Then more in details, the total correlation obtained in CCA is equivalent to a matrix trace norm. And in the trace norm objective layer, the gradient of the trace norm with respect to both text and image features will be computed and back propagated.

2.3 Graph-based methods

Besides DNN-based methods introduced above, there are several other kinds of methods of cross-media retrieval, and one of them is Graph-based method which model the procedure of cross-media retrieval in a graph with vertices and edges. By well defining the meaning of the vertices and the edges in a graph, it can be much easier to apply graph algorithms or methods to deal with the problem of cross-media retrieval.

2.3.1 Example 1: Learning of Multimodal Representations with Random Walks on the Click Graph

Here in this part, we will introduce Graph-based method with an example of the work in [17]. The context of this work is based on the click data collected from the users' searching behavior using a search engine like google, and the search action here is from text query to image results. In this work, they treat the click data as a large click graph, in which vertices are images/text and edges indicate the clicks between an image and a query.

Here is an example of the subgraph of the click graph from a commercial image search engine:

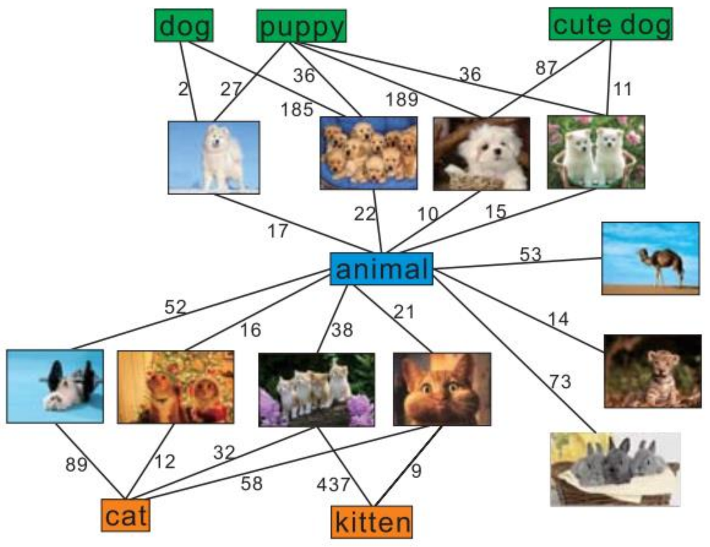


Figure 2.3.1-1 an example of a part of a click graph.

From the figure above, we can see there are two different types of vertices. One type represents textual queries, and the other one represents image search results. As a result, we can easily find out that the graph is a bipartite graph, which means there only can be edges between vertices of different types, and it's not possible to have edges between textual vertices or image vertices. Moreover, Vertices with the same color have the same semantics. And the numbers beside the edges mean that how many clicks on the image search results when searching a specific text query.

After briefly illustrating the graph itself, here is the architecture of the work in the paper which constructs a whole system of cross-media retrieval:

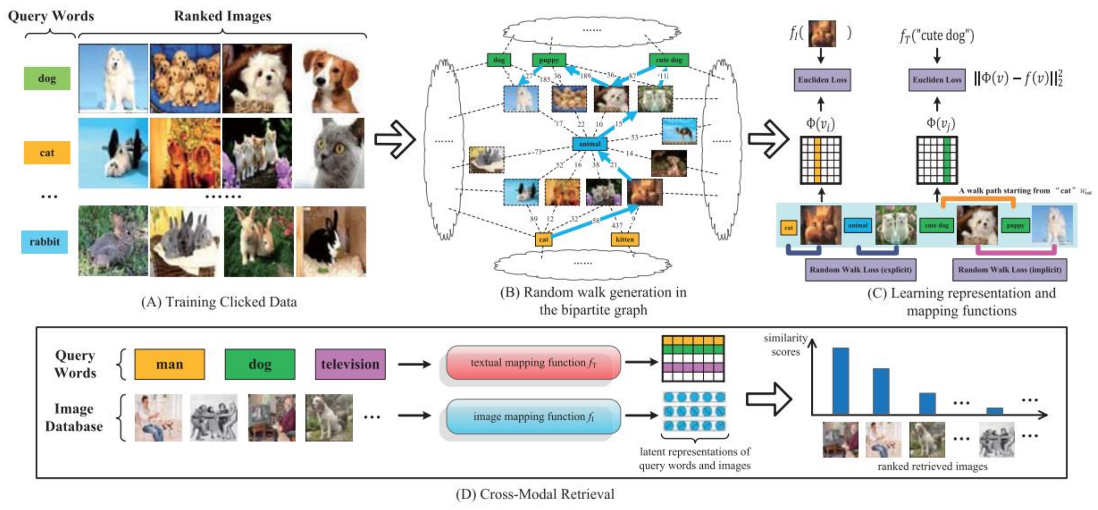

Query Words
Ranked Images
dog
cat
rabbit
(A) Training Clicked Data
(B) Random walk generation in the bipartite graph
(C) Learning representation and mapping functions
Euclidean Loss
f_T("cute dog")
$\|\Phi(v) - f(v)\|_2^2$
$\Phi(v_i)$
$\Phi(v_j)$
A walk path starting from "cat"
Random Walk Loss (explicit)
Random Walk Loss (implicit)
Query Words
Image Database
man
dog
television
textual mapping function f_T
image mapping function f_I
latent representations of query words and images
similarity scores
ranked retrieved images
(D) Cross-Modal Retrieval

Figure 2.3.1-2 an example architecture of graph-based cross-media retrieval system on click graph.

And the work in this paper actually is an extension of 'DeepWalk' proposed in [2.3.1-1]. By modeling the multimodal click graph by a stream of short random walks and adapting techniques of deep neural networks, they present an end-to-end solution, called Multimodal Random Walk Neural Network (MRW-NN) which takes a multimodal click graph as input to learn the common latent representation of text and imagery.

More specifically, there are two phases of the representation learning for each vertex. The first one is the encoding of its context in the graph which is equivalent to learn the neighborhood similarity in its vicinity. And the other one can be called as the internal representation of the vertex which represents the semantics of the content.

And the training process in MRW-NN consists of two stages. First generate random walk path on the click graph with a predefined transition probability. Then given the generated walk paths, the method proposed minimizes the random walk error and the difference between the learned representation and the modality-specific neural network output.

2.4 Learning to Rank methods

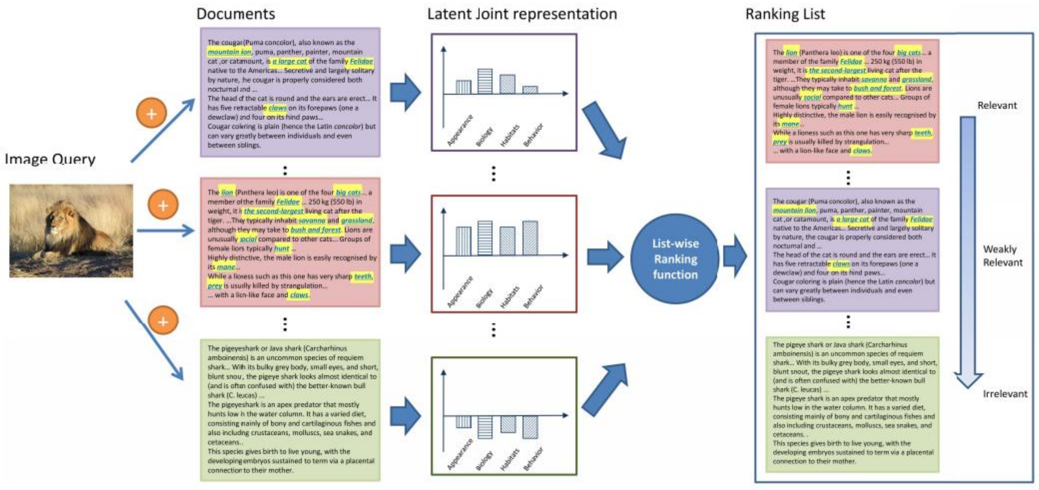
Besides DNN-based methods and graph-based methods, learning to rank methods also can be used to solve the problem of cross-media retrieval, which reformulate the retrieval task as a ranking optimization problem using ranking information as training data. And the research topic of solving cross-media retrieval problem with learning to rank methods also can be called as cross-modal ranking.

2.4.1 Example 1: Cross-Modal Learning to Rank via Latent Joint Representation

Here in this part, we will introduce learning to rank method on cross-media retrieval with an example of the work in [23]. In this paper, an approach which discovers latent joint representations of pairs of cross-media data by using CRF and listwise ranking structural learning. And this approach is named as cross-modal learning to rank via latent joint representation (CML2R).

There is a very big difference between the learned representation of different media type resource in this work and the learned representation in the previous introduced example work. In the previous introduced work in this chapter, resources of different types are both represented in a learned common space, and then the similarity of their representative vectors in the common shared space can be easily compared by means like cosine similarity measure. But in this work, a single latent joint representation is learned for a pair of multi-modal data (e.g., in this work, image query and text document result), which will be used for the latter use of ranking by a list-wise ranking function.

Here is the architecture of the proposed CML2R method:



The cougar(Puma concolor), also known as the mountain lion, puma, panther, painter, mountain cat ,or catamount, is a large cat of the family Felidae native to the Americas... Secretive and largely solitary by nature, he cougar is properly considered both nocturnal and ...
The head of the cat is round and the ears are erect... It has five retractable claws on its forepaws (one a dewclaw) and four on its hind paws...
Cougar colring is plain (hence the Latin concolor) but can vary greatly between individuals and even between siblings.

The lion (Pinthera leo) is one of the four big cats... a member of the family Felidae ... 250 kg (550 lb) in weight, it is the second-largest living cat after the tiger. ...They typically inhabit savanna and grassland, although they may take to bush and forest. Lions are unusually social compared to other cats... Groups of female lions typically hunt ...
Highly distinctive, the male lion is easily recognised by its mane...
While a lioness such as this one has very sharp teeth, prey is usuilly killed by strangulation...
... with a lion-like face and claws.

The pigeyeshark or Java shark (Carcharhinus amboinensis) is an uncommon species of requiem shark... With its bulky grey body, small eyes, and short, blunt snou , the pigeye shark looks almost identical to (and is often confused with) the better-known bull shark (C. leucas) ...
The pigeyeshark is an apex predator that mostly hunts low in the water column. It has a varied diet, consisting mainly of bony and cartilaginous fishes and also includng crustaceans, molluscs, sea snakes, and cetaceans..
This species gives birth to live young, with the developingembryos sustained to term via a placental connection to their mother.

Figure 2.4.1 the architecture of CML2R work.

More in details, they use a method from probabilistic graph model named Conditional Random Field (CRF) to learn the mentioned. And in the phase of training the parameters for both CRF and the designed list-wise ranking function, the training problem can be regarded as solving a standard structural-SVM, which can be solved by their proposed optimization method.

2.5 Hashing methods

As we know, hashing usually can be used as encoding to represent and compress various kinds of information. And here in this chapter, we will introduce how to cope with cross-media retrieval using hashing methods. And in order to do that, we should find a way to learn some specific hashing functions to encode resource of different types, such as images and text. As a result, different hashing-based methods mainly differ in the aspect of the way to design and obtain different hashing functions.

2.5.1 Example 1: Deep Discrete Cross-Modal Hashing for Cross-Media Retrieval

This example is the work of [25], which will be introduced here as an example of hashing method used to solve cross-media retrieval. Since there is the heterogeneity gap among various modality data, such as texts and images, which makes it difficult to perform neighbor-based method like Approximate Nearest Neighbor (ANN) search. Moreover, the cross-media retrieval in large-scale and high-dimensional datasets can be really challenging since the high cost of storage and computational complexity. As a result, the reason why hashing becomes more and more attractive for researchers is that it can be effective in reducing storage and computational cost. And the goal of hashing is to learn hash functions that map high-dimensional data to a Hamming space in which data are encoded as compact binary codes. Then the similarities between multi-modal data can be measured by Hamming distances in a common Hamming space.

To solve the challenges mentioned above, the authors proposed a discrete cross-media hashing based on deep neural network which is named as Deep Discrete Cross-Modal Hashing (DDCMH). And in this work, DDCMH learns the compact 8-bit binary codes directly by formulating the problem as a quantization optimization with the final object function. Thus, DDCMH is a new framework of cross-modal DNNs to seek multiple hierarchical non-linear transformations to jointly learn compact binary codes and non-linear hash functions.

Here is the architecture of the proposed DDCMH framework:

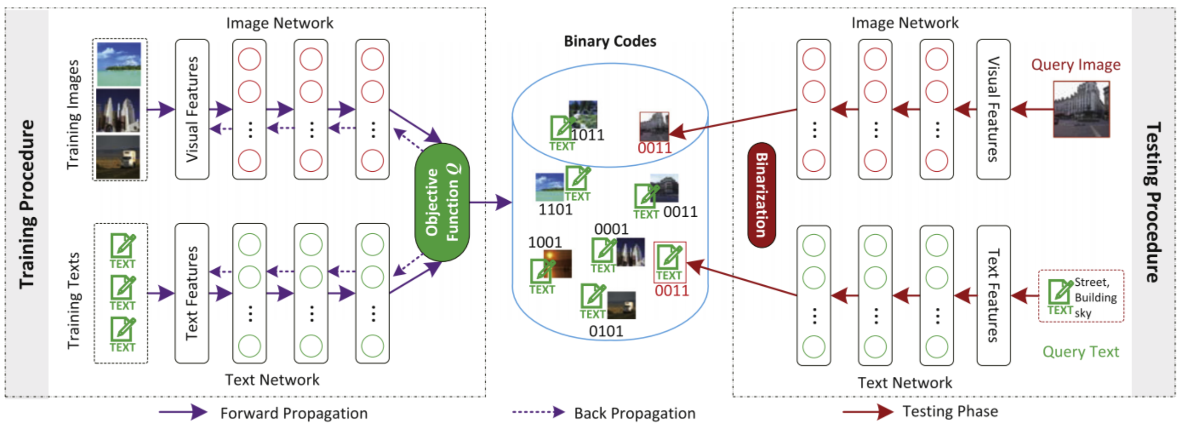


Figure 2.5.1 the architecture of DDCMH framework.

As the figure shows above, they develop a novel framework with two deep neural networks which are trained as hash functions to learn binary codes for image and text modality respectively. Different from linear transformations, such cross-modal deep neural networks are able to well capture the non-linear relationship in each modality. And this work embeds the intra-modality similarity preservation into every hidden layer of each modality. In the meanwhile, the inter-modality preservation is formulated between the outputs of two deep networks at the top layers.

2.6 NLP-based methods

As the work of BERT [2.6.1-1] became an explosion point in late 2018 and brought another age of bloom of Natural Language Processing, it has been more and more popular as the representation of text information among researchers of NLP and relevant fields. Moreover, the work of Transformer [2.6.1-2] itself which the previously mentioned BERT is based on can also be a strong representation of text. As a result, more and more NLP-based methods with BERT or Transformer have been proposed to solve the problem of cross-media retrieval.

2.6.1 Example 1: Multi-modal Transformer for Video Retrieval

This work is from [28], and it deals with the task of retrieving video content relevant to natural language queries. And in this paper, the authors propose a multi-model Transformer to jointly encode the different modalities in video which is also able to encode the temporal information in video. On the NLP side,
after investigating different representative methods, BERT has been chosen to represent text or caption. Finally, a weighted similarity measure has been proposed for the ranking of retrieval.

And here is the architecture of the proposed framework with Multi-modal Transformer (MMT):

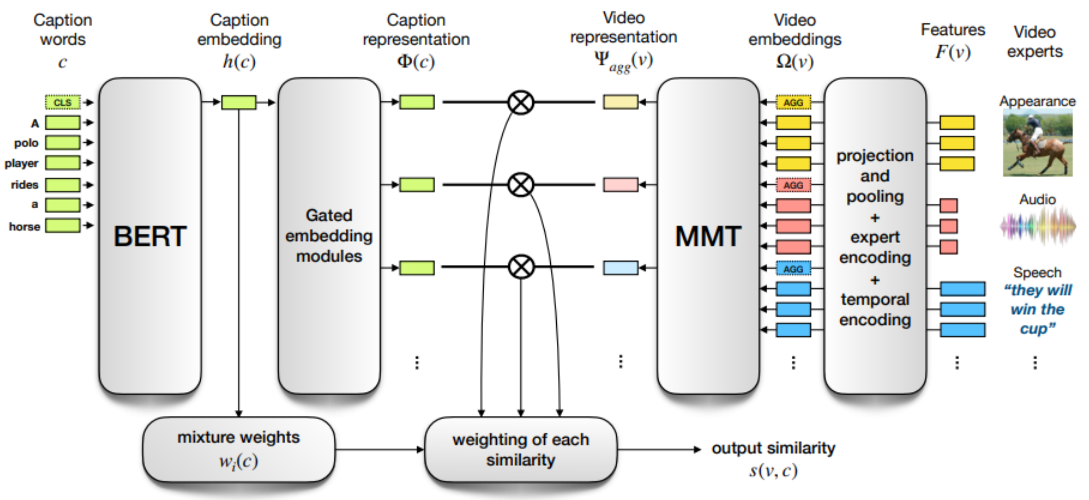

Caption words c
Caption embedding $h(c)$
Caption representation $\Phi(c)$
Video representation $\Psi_{agg}(v)$
Video embeddings $\Omega(v)$
Features $F(v)$
Video experts
CLS
A
polo
player
rides
a
horse
BERT
Gated embedding modules
MMT
AGG
projection and pooling + expert encoding + temporal encoding
Appearance
Audio
Speech
"they will win the cup"
mixture weights $w_i(c)$
weighting of each similarity
output similarity $s(v,c)$

Figure 2.6.1-1 the architecture of Multi-modal Transformer (MMT) framework.

As shown in the figure above, we can see that on the left side, the representation of the caption is based on BERT and a following gated embedding module. And on the right side, there are three different parts of video information, which are called Features, Expert embeddings and Temporal embeddings, through the proposed Multi-modal Transformer, finally the video representation is obtained.

And here is the three parts of video embeddings:

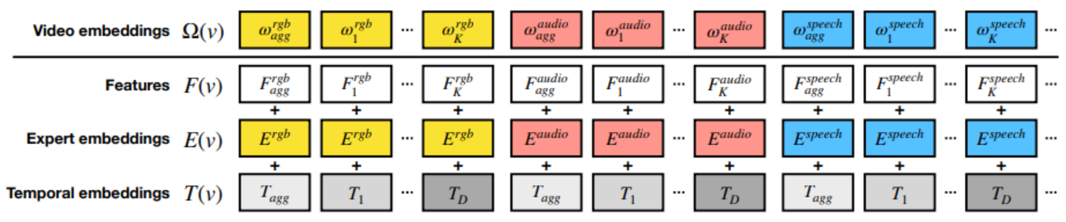

Video embeddings $\Omega(v)$
ω^{rgb}_{agg} ω^{rgb}_{1} ω^{rgb}_{K} ω^{audio}_{agg} ω^{audio}_{1} ω^{audio}_{K} ω^{speech}_{agg} ω^{speech}_{1} ω^{speech}_{K}
Features $F(v)$
F^{rgb}_{agg} F^{rgb}_{1} F^{rgb}_{K} F^{audio}_{agg} F^{audio}_{1} F^{audio}_{K} F^{speech}_{agg} F^{speech}_{1} F^{speech}_{K}
Expert embeddings $E(v)$
E^{rgb} E^{audio} E^{speech}
Temporal embeddings $T(v)$
T_{agg} T_1 T_D

Figure 2.6.1-2 Inputs to the multi-modal transformer.

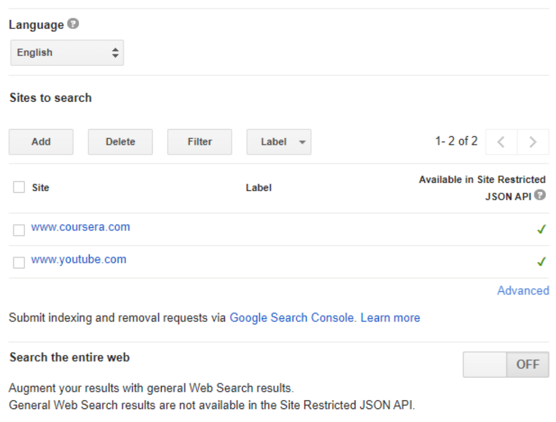
Chapter 3

Algorithms and tools utilized

3.1 google custom search

There is a very cool function provided by Google which is called 'google custom search'. And it allows people customizing their search procedure by allowing people defining a bunch of specific websites from which the search results are retrieved. For example, if we only want to get search results from two websites, YouTube and Coursera, then we define the set of search websites with these two websites above. As a result, if we search 'machine learning' in our own custom search engine, we will get all the relevant videos only from the two websites mentioned above.

And this functionality of google custom search can be really helpful for us, since we can take advantage of the customized search engine as a part of the pipeline of our project. And here a brief introduction will be done.


Language
English

Sites to search

Site	Label	Available in Site Restricted JSON API
www.coursera.com		✓
www.youtube.com		✓

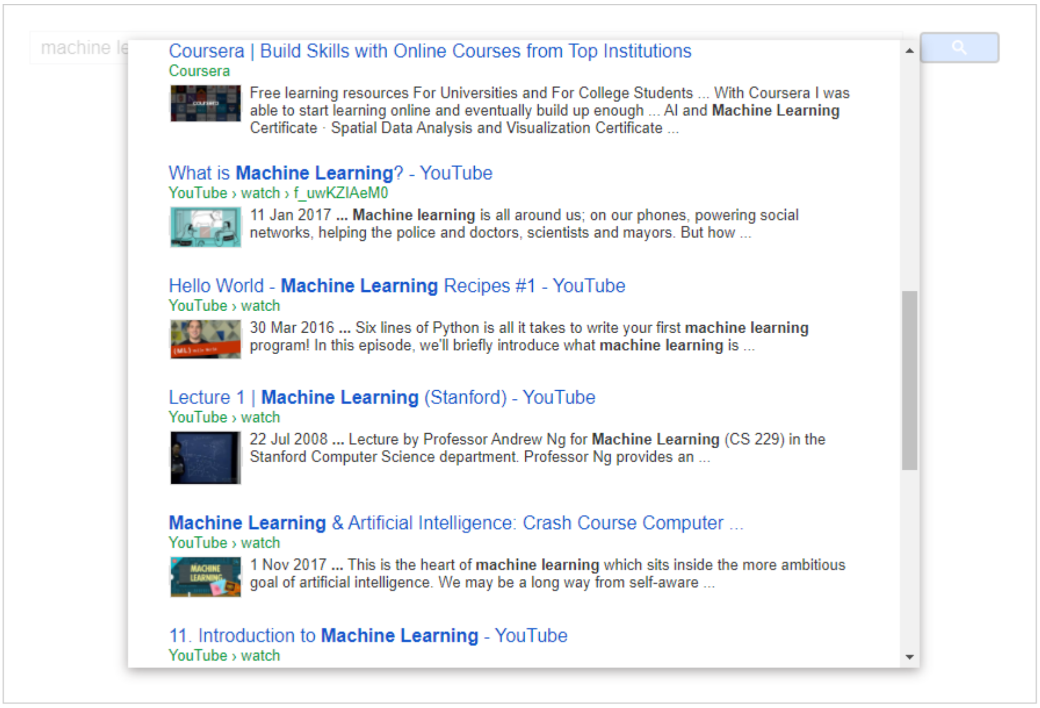
Submit indexing and removal requests via Google Search Console. Learn more

Search the entire web
OFF

Augment your results with general Web Search results.
General Web Search results are not available in the Site Restricted JSON API.

Figure 3.1 Set specific web sites to be customize searched on.

The figure above shows that we set two websites: Coursera and YouTube as the target websites in our custom search engine.


machine le

Coursera | Build Skills with Online Courses from Top Institutions
Coursera
Free learning resources For Universities and For College Students ... With Coursera I was able to start learning online and eventually build up enough ... AI and **Machine Learning** Certificate · Spatial Data Analysis and Visualization Certificate ...

What is **Machine Learning**? - YouTube
YouTube › watch › f_uwKZIAeM0
11 Jan 2017 ... **Machine learning** is all around us; on our phones, powering social networks, helping the police and doctors, scientists and mayors. But how ...

Hello World - **Machine Learning** Recipes #1 - YouTube
YouTube › watch
30 Mar 2016 ... Six lines of Python is all it takes to write your first **machine learning** program! In this episode, we'll briefly introduce what **machine learning** is ...

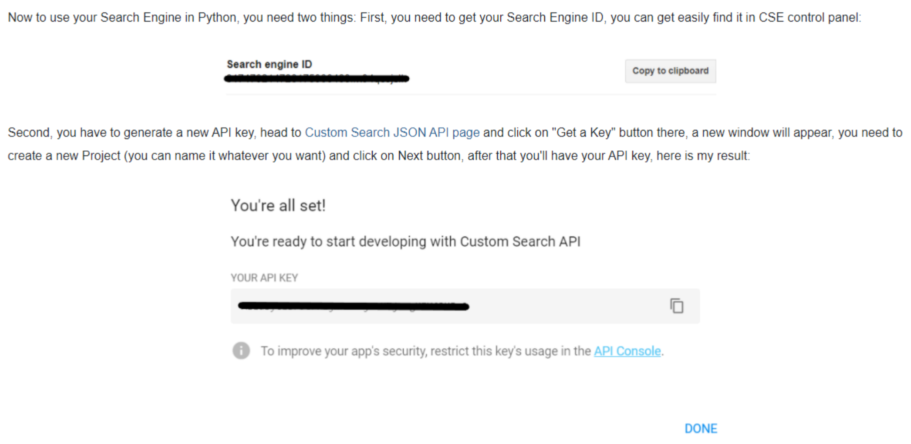
Lecture 1 | **Machine Learning** (Stanford) - YouTube
YouTube › watch
22 Jul 2008 ... Lecture by Professor Andrew Ng for **Machine Learning** (CS 229) in the Stanford Computer Science department. Professor Ng provides an ...

Machine Learning & Artificial Intelligence: Crash Course Computer ...
YouTube › watch
1 Nov 2017 ... This is the heart of **machine learning** which sits inside the more ambitious goal of artificial intelligence. We may be a long way from self-aware ...

11. Introduction to **Machine Learning** - YouTube
YouTube › watch

Figure 3.2 Search results on query “machine learning” using google custom search

moreover, there is also the Custom Search JSON API that enable us developing applications to retrieve search results from google search engine customizely. And we can get all kinds of available search results such as documents, images by using RESTFUL requests to this API. But before using this API in your own program, you should first set an API key:


Now to use your Search Engine in Python, you need two things: First, you need to get your Search Engine ID, you can get easily find it in CSE control panel:

Search engine ID
[redacted]

Second, you have to generate a new API key, head to Custom Search JSON API page and click on "Get a Key" button there, a new window will appear, you need to create a new Project (you can name it whatever you want) and click on Next button, after that you'll have your API key, here is my result:

You're all set!

You're ready to start developing with Custom Search API

YOUR API KEY
[redacted]

To improve your app's security, restrict this key's usage in the API Console.

Figure 3.3 Set google custom search API key for invocation from outside.

After obtaining the google custom search ID and the API key. Then we will use this API with some specific programming languages, such as python, java, etc. In order to use the google custom search API in python, we should first install the google-api-python-client. The installation can be done using pip as below:

pip install google-api-python-client

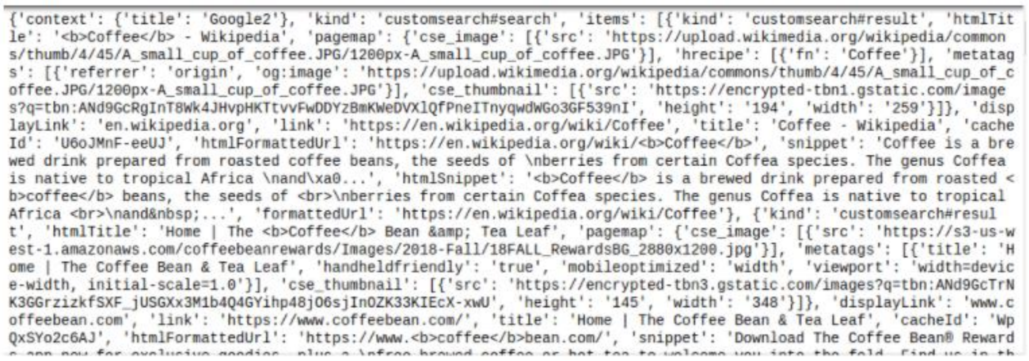
After the installation, then we are able to import the google-api-python-client in the program. Then the main usage of the API can be implemented as below:

```
from googleapiclient.discovery import build
my_api_key = "Your API Key"
my_cse_id = "Your CSE ID"

def google_search(search_term, api_key, cse_id, **kwargs):
    service = build("customsearch", "v1", developerKey=api_key)
    res = service.cse().list(q=search_term, cx=cse_id, **kwargs).execute()
    return res
```

In the code shown above, we should set the value of my_api_key and my_cse_id variables as the API key and the google custom search engine ID respectively. Then the only thing left to be done is to invoke the function created above with a specific query.

```
result = google_search("Coffee", my_api_key, my_cse_id)
print(result)
```

As a result, the invoke of the function above will search the input query and return the relevant documents indexed in the search engine. And an object in Json format will be returned which make it possible for latter parse of the detail information of the search result:



```
{'context': {'title': 'Google2'}, 'kind': 'customsearch#search', 'items': [{'kind': 'customsearch#result', 'htmlTit
le': '<b>Coffee</b> - Wikipedia', 'pagemap': {'cse_image': [{'src': 'https://upload.wikimedia.org/wikipedia/common
s/thumb/4/45/A_small_cup_of_coffee.JPG/1200px-A_small_cup_of_coffee.JPG'}], 'hrecipe': [{'fn': 'Coffee'}], 'metatag
s': [{'referrer': 'origin', 'og:image': 'https://upload.wikimedia.org/wikipedia/commons/thumb/4/45/A_small_cup_of_c
offee.JPG/1200px-A_small_cup_of_coffee.JPG'}], 'cse_thumbnail': [{'src': 'https://encrypted-tbn1.gstatic.com/image
s?q=tbn:ANd9GcRgInT8Wk4JHvpHKTtvvFwDDYzBmKWeDVXlQfPneITnyqwdWGo3GF539nI', 'height': '194', 'width': '259'}]}, 'disp
layLink': 'en.wikipedia.org', 'link': 'https://en.wikipedia.org/wiki/Coffee', 'title': 'Coffee - Wikipedia', 'cache
Id': 'U6oJMnF-eeUJ', 'htmlFormattedUrl': 'https://en.wikipedia.org/wiki/<b>Coffee</b>', 'snippet': 'Coffee is a bre
wed drink prepared from roasted coffee beans, the seeds of \nberries from certain Coffea species. The genus Coffea
is native to tropical Africa \nand\xa0...', 'htmlSnippet': '<b>Coffee</b> is a brewed drink prepared from roasted <
b>coffee</b> beans, the seeds of <br>\nberries from certain Coffea species. The genus Coffea is native to tropical
Africa <br>\nand&nbsp;...', 'formattedUrl': 'https://en.wikipedia.org/wiki/Coffee'}, {'kind': 'customsearch#resul
t', 'htmlTitle': 'Home | The <b>Coffee</b> Bean &amp; Tea Leaf', 'pagemap': {'cse_image': [{'src': 'https://s3-us-w
est-1.amazonaws.com/coffeebeanrewards/Images/2018-Fall/18FALL_RewardsBG_2880x1200.jpg'}], 'metatags': [{'title': 'H
ome | The Coffee Bean & Tea Leaf', 'handheldfriendly': 'true', 'mobileoptimized': 'width', 'viewport': 'width=devic
e-width, initial-scale=1.0'}], 'cse_thumbnail': [{'src': 'https://encrypted-tbn3.gstatic.com/images?q=tbn:ANd9GcTrN
K3GGrzizkfSXF_jUSGXx3M1b4Q4GYihp48jO6sjInOZK33KIEcX-xwU', 'height': '145', 'width': '348'}]}, 'displayLink': 'www.c
offeebean.com', 'link': 'https://www.coffeebean.com/', 'title': 'Home | The Coffee Bean & Tea Leaf', 'cacheId': 'Wp
QxSYo2c6AJ', 'htmlFormattedUrl': 'https://www.<b>coffee</b>bean.com/', 'snippet': 'Download The Coffee Bean® Reward
```

Figure 3.4 Example of return results using google custom search in python.

The JSON object returned above is very similar to the result from the Google search. And obtaining information from a customized google search engine isn't that hard. Since the Custom Search API makes everything easy, as the only difficulty is in parsing the JSON object for the needed information.

3.2 PyPDF

PyPDF is a PDF toolkit launched by Mathieu Fenniak in 2005, which is focused on:

•document manipulation: by-page splitting, concatenation, and merging;

•document introspection;

•page cropping; and

•document encryption and decryption.

Here is a brief introduction of beginning using PyPDF in python.

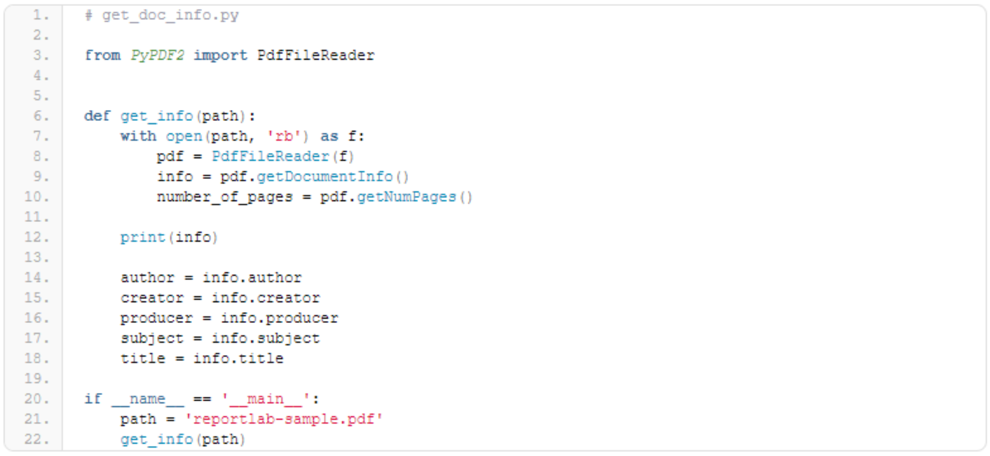
Step 1. Installation: PyPDF2 is a pure Python package, so you can install it using pip:

```
python -m pip install pypdf2
```

As usual, you should install 3rd party Python packages to a Python virtual environment to make sure that it works the way you want it to.

Step 2. Extract Metadata from PDFs:

PyPDF has been one of the most popular tools to extract useful meta-data from any PDF. The meta-data of a PDF usually consists of the author, title, number of pages and contents of the PDF. Here is an example of using PyPDF to extract meta-data of a specific book:



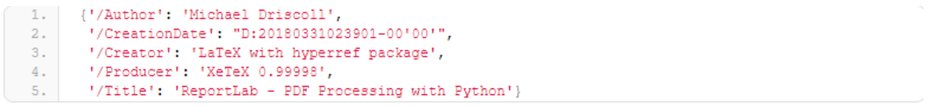
```python
# get_doc_info.py

from PyPDF2 import PdfFileReader


def get_info(path):
    with open(path, 'rb') as f:
        pdf = PdfFileReader(f)
        info = pdf.getDocumentInfo()
        number_of_pages = pdf.getNumPages()

    print(info)

    author = info.author
    creator = info.creator
    producer = info.producer
    subject = info.subject
    title = info.title

if __name__ == '__main__':
    path = 'reportlab-sample.pdf'
    get_info(path)
```

Figure 3.5 Extraction of metadata from PDF using PyPDF.

There is a class called PdfFileReader in PyPDF which enable us to read a PDF and extract meta-data in it. And in the example code above, a function called get_info has been made to make use of PdfFileReader. At first, we open that file of the PDF and put the file handler into PdfFileReader to create an object of it. Then we are able to extract meta-data of the PDF by using the getDocumentInfo method. The returned result of the method is an instance of PyPDF2.pdf.DocumentInformation class, which consists of the following attributes:

- author
- creator
- producer
- subject
- title

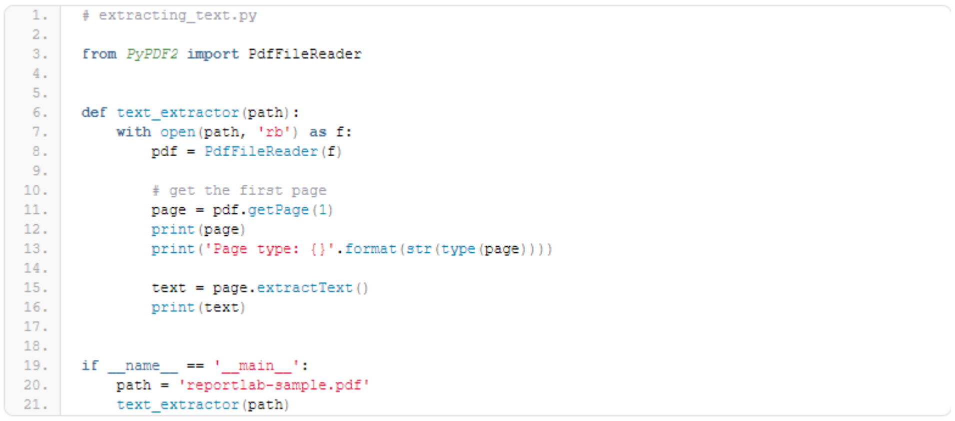
Here is an example of an DocumentInformation object:



```
{'/Author': 'Michael Driscoll',
 '/CreationDate': "D:20180331023901-00'00'",
 '/Creator': 'LaTeX with hyperref package',
 '/Producer': 'XeTeX 0.99998',
 '/Title': 'ReportLab - PDF Processing with Python'}
```

Step 3. Extract text from PDF:

Finally, here is an example to extract the text of the a specific page of the PDF prepared before:



```python
# extracting_text.py

from PyPDF2 import PdfFileReader


def text_extractor(path):
    with open(path, 'rb') as f:
        pdf = PdfFileReader(f)

        # get the first page
        page = pdf.getPage(1)
        print(page)
        print('Page type: {}'.format(str(type(page))))

        text = page.extractText()
        print(text)


if __name__ == '__main__':
    path = 'reportlab-sample.pdf'
    text_extractor(path)
```

Figure 3.6 Extraction of text from PDF using PyPDF.

3.3 Named entity recognition and linking

Named-entity recognition (NER) belongs to the field of information extraction which targets to locate and classify named entities presented in raw text into pre-defined classes such as names of specific persons, organizations, or locations, etc.

Usually there are two phases in the procedure of NER: 1. detection of names or labels of named entities. 2. classification of the detected named entities into a specific pre-defined type or class (i.e. person, organization, location, etc.). The first phase in practice can be seen as a problem of segmentation: names of named entities are usually several contiguous words mentioned in text. For example, "Bank of Italy" should be a single name of named entity, even inside this single name, the strings "Bank" and “Italy” themselves are also names. And the second phase does a classification work according to a pre-selected ontology or categorical system.

In the field of NLP, entity linking has several different names, such as named-entity linking (NEL), named-entity disambiguation (NED), named-entity recognition and disambiguation (NERD) with the same meaning. And NEL is target to specify the unique identity of entities (such as named people, famous locations, or companies) appear in text. As an example, the sentence "Let’s go to the apple store in the city center", the target of NEL is to assert that "apple" refers to the famous international technology company and not to the fruit of apple. And the main difference between NEL and NER is that NER only recognize the appearance of a specific named entity in text but it does not find out the specific entity URI.

In the domain of NEL, the special words appear in text are retrieved from a source of named entities called knowledge base. And special words mentioned above are named as mentions, surfaces or labels of named entities. Usually the used knowledge base should be chosen according to the specific motivate, and for entity linking applications which work on open-fields knowledge bases such as wikidata or DBpedia have always been the most popular selections, since they are both derived from wikepedia.

As already introduced above, the URIs of named entities uniquely identify named entities in a specific knowledge base. But there will be a natural problem of that for a unique named entity the URI in different knowledge bases can be absolutely different. But don't be worried too much, there is a mapping of named entity URIs between the most popular knowledge bases built from Wikipedia such as wikidata and DBpedia.

NEL plays an important role in bridging raw text data in web with pre-produced knowledge bases, which helps a lot for the annotation of tremendous quantity of raw text data with noise on the internet and makes great contributions to the development of Semantic web. What's more, NEL can be quite beneficial in domains in which the extraction of semantic representations in text is useful, such as analysis on text contents, recommendation systems, search engines and chatbots.

As an example, the main task of search engines is to check out the documents which are most relevant to a specific given text query. Then if a given query is "Who is the current president of united states", NEL should be used to recognize the specific named entities mentioned in the query, such as president and united states, then it can be much easier for the search engine to return reasonable documents which answer the question in the query properly.

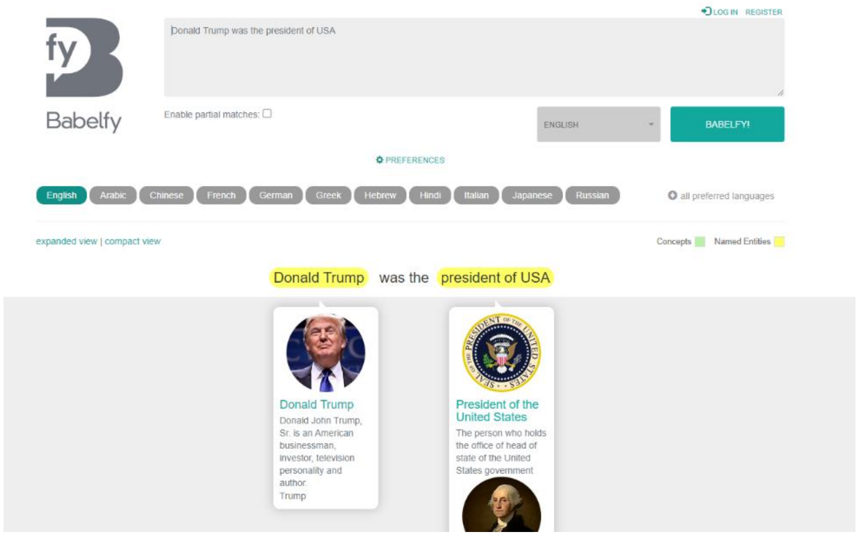
There are many different implemented tools to do the NERD task, such as Babelfy, DBpedia Spotlight, and AIDA. Here we briefly introduce the tools mentioned above.

Babelfy is a multi-lingual text disambiguation software. And the two main functions of Babelfy are word sense disambiguation and entity linking. BabelNet multilingual semantic network is the base of Babelfy. And Babelfy does the work of NEL in the following steps:

- It assigns a semantic signature to each vertex of the BabelNet semantic network (i.e. concept or named entity). And this step should be performed only once, which has no matter with the input text.

- Then it extracts all the linkable segments from the input text. And for each of the linkable segment, it lists all the possible meaning from the semantic network.

- Next, it creates a semantic representation of the whole text by linking the candidate meanings of the linkable segments with the semantic signatures previously-computed. Finally, it extracts a subgraph of the representation created above and chooses the best candidate meaning for each segment.



Donald Trump was the president of USA

Donald Trump was the president of USA

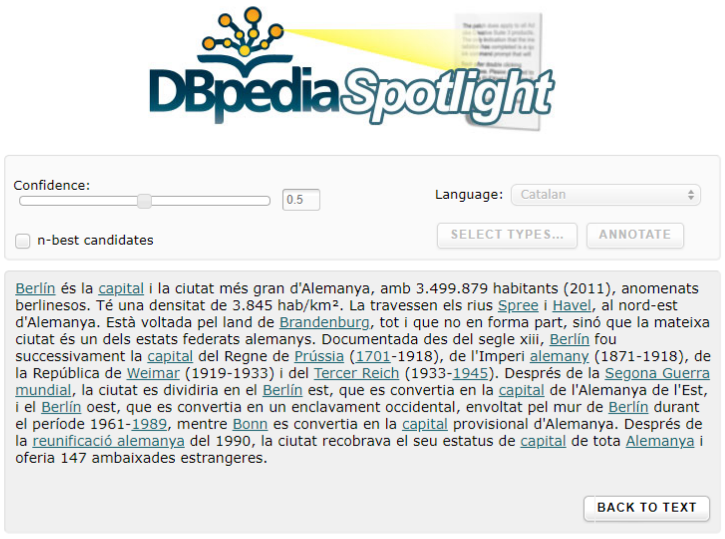
Donald Trump
Donald John Trump, Sr. is an American businessman, investor, television personality and author.
Trump

President of the United States
The person who holds the office of head of state of the United States government

Figure 3.7 Example of named entity linking using Babelfy.

DBpedia Spotlight is a tool for annotating mentions of DBpedia resources in natural language text, providing capabilities useful for Named Entity Recognition, Name Resolution, amongst other information extraction tasks.

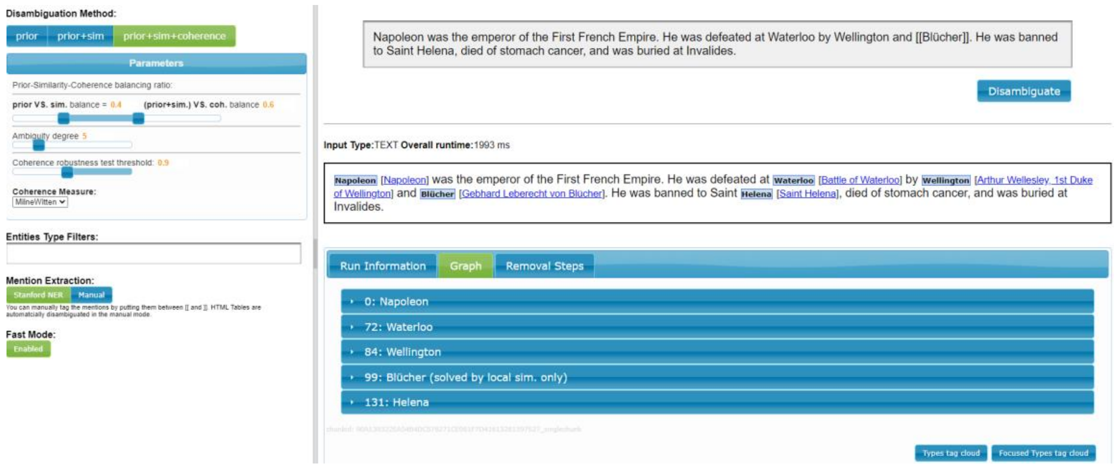
DBpedia Spotlight allows users to configure the annotations to their specific needs through the DBpedia Ontology and quality measures such as prominence, topical pertinence, contextual ambiguity and disambiguation confidence.



Berlín és la capital i la ciutat més gran d'Alemanya, amb 3.499.879 habitants (2011), anomenats berlinesos. Té una densitat de 3.845 hab/km². La travessen els rius Spree i Havel, al nord-est d'Alemanya. Està voltada pel land de Brandenburg, tot i que no en forma part, sinó que la mateixa ciutat és un dels estats federats alemanys. Documentada des del segle xiii, Berlín fou successivament la capital del Regne de Prússia (1701-1918), de l'Imperi alemany (1871-1918), de la República de Weimar (1919-1933) i del Tercer Reich (1933-1945). Després de la Segona Guerra mundial, la ciutat es dividiria en el Berlín est, que es convertia en la capital de l'Alemanya de l'Est, i el Berlín oest, que es convertia en un enclavament occidental, envoltat pel mur de Berlín durant el període 1961-1989, mentre Bonn es convertia en la capital provisional d'Alemanya. Després de la reunificació alemanya del 1990, la ciutat recobrava el seu estatus de capital de tota Alemanya i oferia 147 ambaixades estrangeres.

Figure 3.8 Example of named entity linking using DBpedia Spotlight.

AIDA is another framework for NEL. Given a text in natural language, it links words mentioned to named entities such as famous people, places and organizations, which have been collected in a specific knowledge base such as DBpedia, Freebase, and YAGO. AIDA is a strong framework which is based on comparing similarity between the context of a mention and its candidates named entities.



Prior-Similarity-Coherence balancing ratio:

prior VS. sim. balance = 0.4 (prior+sim.) VS. coh. balance 0.6

Ambiguity degree 5

Coherence robustness test threshold: 0.9

Coherence Measure: MilneWitten

Entities Type Filters:

Mention Extraction:

You can manually tag the mentions by putting them between [[and]]. HTML Tables are automatcially disambiguated in the manual mode.

Fast Mode:

Napoleon was the emperor of the First French Empire. He was defeated at Waterloo by Wellington and [[Blücher]]. He was banned to Saint Helena, died of stomach cancer, and was buried at Invalides.

Input Type:TEXT Overall runtime:1993 ms

Napoleon [Napoleon] was the emperor of the First French Empire. He was defeated at Waterloo [Battle of Waterloo] by Wellington [Arthur Wellesley, 1st Duke of Wellington] and Blücher [Gebhard Leberecht von Blücher]. He was banned to Saint Helena [Saint Helena], died of stomach cancer, and was buried at Invalides.

0: Napoleon
72: Waterloo
84: Wellington
99: Blücher (solved by local sim. only)
131: Helena

Figure 3.9 Example of named entity linking using AIDA.

3.4 front-end languages

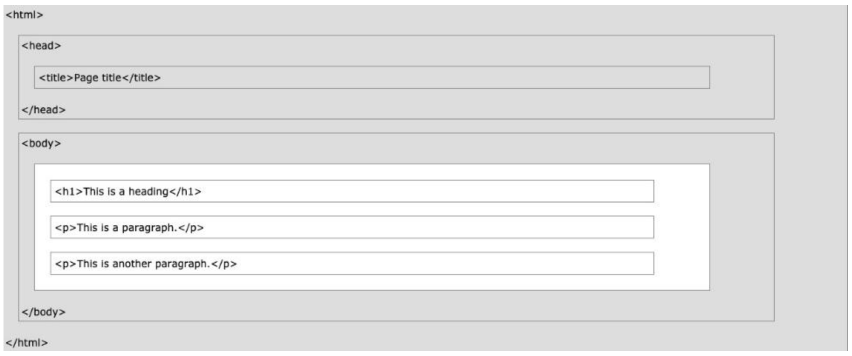
In order to link the html book and the related video resources together, the front-end tools of HTML, CSS, JavaScript are used here.

1. HTML to define the structure of web pages
2. CSS to control the outlook of web pages
3. JavaScript to define the behavior of web pages

To put it in another way, the process of forming a web page is similar to build a house, HTML is used to construct the structure, then CSS does the makeup job like putting up the curtains and installing floor. After all that JavaScript designs the logic system as the electric system, which guarantee the house is ready to live in with all living functions.

3.4.1 HTML

HTML is the short-name of Hyper Text Markup Language. In fact, it can't be classified as a programming language, it's actually a standard markup language that is dedicated to creating Web pages. And nowadays in most of cases the structures of a Web pages are descripted by HTML. A series of elements have been organized together to instruct the browser the way to display the content. And different types of contents are indicated by using different kinds of elements such as "header", "paragraph", and "link", etc.



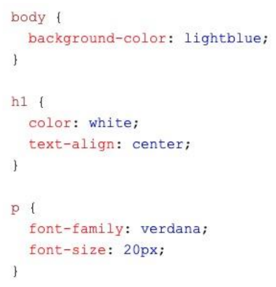
```
<html>
  <head>
    <title>Page title</title>
  </head>
  <body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
    <p>This is another paragraph.</p>
  </body>
</html>
```

Figure 3.10 Example of HTML language.

3.4.2 CSS

CSS stands for Cascading Style Sheets. In HTML files the element classes are defined, the content is sorted as the distributed classes. Here in CSS file, it describes how HTML elements are to be displayed on webpage of computer screen, paper, or on the mobile phones, like defining the font size of paragraph as 20px, the font family as verdana, the background color of body element as light blue, the text alignment of heads as in center, the position of an element as a certain place.

To compare with writing CSS definition in HTML, CSS saves a lot of work by controlling the layout of multiple web pages all at once. Also, external stylesheets are stored in CSS files, which makes the HTML codes less and cleaner. By defining different classes, it is easy to control the different styles of all elements of whole program.



```
body {
  background-color: lightblue;
}

h1 {
  color: white;
  text-align: center;
}

p {
  font-family: verdana;
  font-size: 20px;
}
```

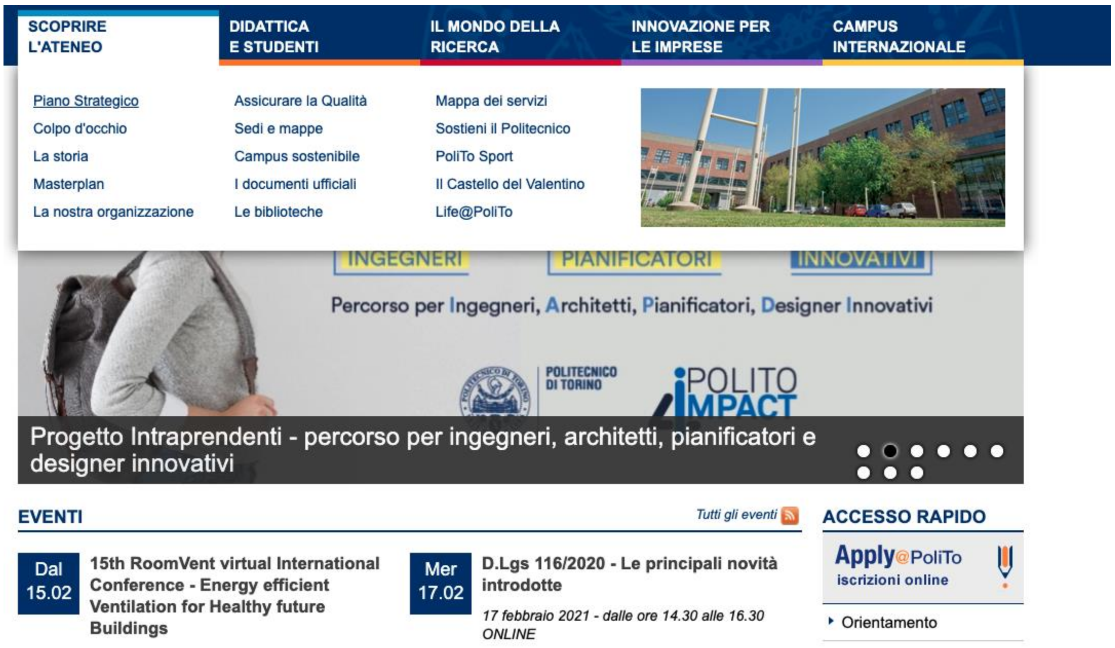
Figure 3.11 Example of CSS language.

3.4.3 JavaScript

JavaScript (JS) is a programming language which is lightweight and interpreted with first-class functions, so that the functions in that language are treated the same as normal variables. For instance, in JavaScript, a function can be used as an argument or a return value of another function.

A webpage defined by HTML and CSS can only be read by user. With JavaScript, the interactions of user and the webpage could be realized, the behaviors of client-end present more cool effects. The functions of second level navigation, the display in turn of images or the back to top are all realized by JavaScript.

In JavaScript file, the content of HTML can be changed, the attribute values can be justified. Also, the CSS style could be defined again. For example, the button text is defined in HTML, CSS makes sure how the button looks like, here in JavaScript, the function of the button is defined, like which part is this button linked to, also the style of button when the mouse is hanging on or after clicking can be indicated.



Progetto Intraprendenti - percorso per ingegneri, architetti, pianificatori e designer innovativi

Dal 15.02 15th RoomVent virtual International Conference - Energy efficient Ventilation for Healthy future Buildings

Mer 17.02 D.Lgs 116/2020 - Le principali novità introdotte

17 febbraio 2021 - dalle ore 14.30 alle 16.30 ONLINE

Figure 3.12 The home page of Polito is created using front-end languages.

3.5 Machine Learning algorithms

In order to improve the search result consisting of relevant YouTube videos according to a specific query. Different kinds of machine learning models have been tried to do a classification between the labeled good results and labeled bad results. Here we only introduce two of them briefly.

3.5.1 Logistic Regression

Logistic Regression is one of the most popular machine learning algorithms that has been widely used for the work of classification. And classification can be addressed as a job of determination of the category or class of a specific sample by using the features of that sample. Here is a picture which shows the meaning above.

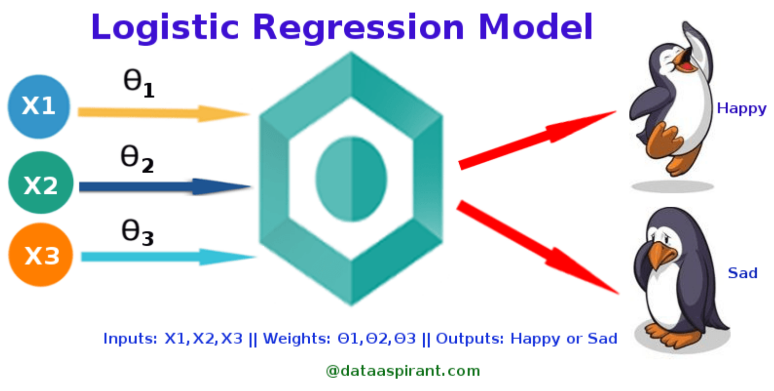

Logistic Regression Model
X1
X2
X3
θ1
θ2
θ3
Happy
Sad
Inputs: X1,X2,X3 || Weights: Θ1,Θ2,Θ3 || Outputs: Happy or Sad
@dataaspirant.com

Figure 3.13 A direct representation of logistic regression classifier.

Logistic regression is a learning method which learns the classification probabilities with an input feature vector. And Logistic regression can either do a binary classification or a multinomial classification in which there are not only two different classes.

More in details, Logistic regression belongs to linear model since the logit form of the model is linear, and logit means the log of odds of a specific probabilistic event. And Logistic regression uses maximum likelihood to estimate the parameters of the model. What's more, Logistic regression is only a special case of generalized linear model when the response variable follows the binomial distribution. And here are some figures with the relevant mathematic representations of Logistic regression.

$$log\left(\frac{P(y=1)}{1-P(y=1)}\right) = log\left(\frac{P(y=1)}{P(y=0)}\right) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

$$\frac{P(y=1)}{1-P(y=1)} = odds = exp\,(\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p)$$

3.5.2 Xgboost

Xgboost is both an ensemble machine learning method and an open-source software library which provides a regularizing gradient boosting framework. It is an improved version of a tree-based ensemble machine learning method GBDT, and it has gained much popularity and attention recently as the algorithm of choice for many winning teams of machine learning competitions.

For a brief address of the theory part of Xgboost, it belongs to a family of ensemble machine learning algorithms called boosting, which means train series of base classifiers each step to improve the classification result of the current ensemble classifier.

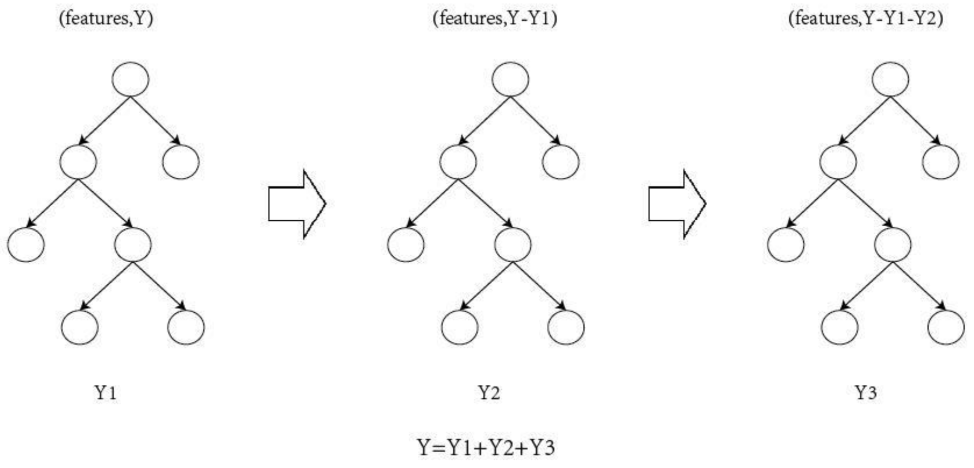


Figure 3.14 A direct representation of Xgboost classifier.

The figure above shows both the architecture and the process of training of xgboost. In most of times, the base classifier is a decision tree. Then xgboost train a different decision tree at each step to approximate the residual between the previous ensemble model and the ground-truth.

And the figure below shows some of the main characteristics of Xgboost:

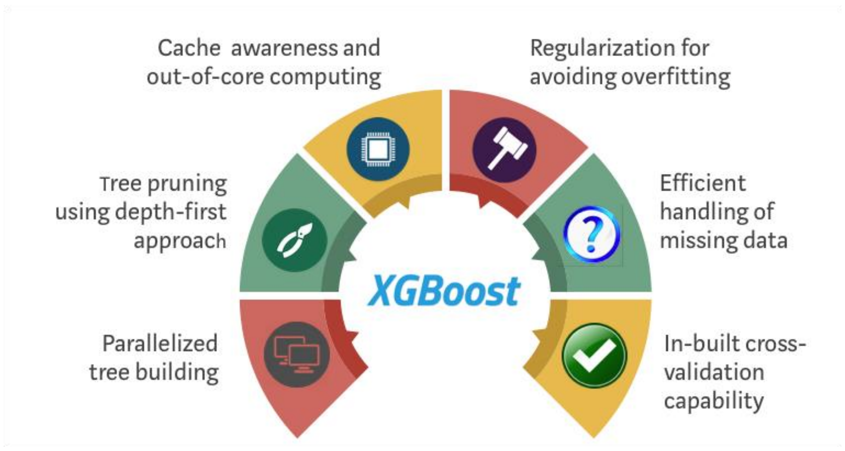

Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

Figure 3.15 Main characteristics of Xgboost.

3.6 Mean average precision

In the domain of information retrieval, there are different metrics that assess the capacity of a certain search engine, such as precision, recall, etc. But there is a big defect of precision and recall as single-value metrics calculated on the whole list of returned documents from a classification or ranking system. If the system belongs to ranking system which returns a ranking list of documents, it is natural that the order of the returned documents should also be considered.

3.6.1 Average precision

A plot of precision-recall curve can be created by computing the values of precision and recall at every position in the ranking list of returned documents. Then we regard precision p(r) as a function of recall r. As a result, average precision(ap) can be defined and calculated as the mean of p(r) over the domain of r=0 to r=1.

$$\text{AveP} = \int_0^1 p(r)dr$$

We can easily see that ap is just the area under the precision-recall curve. And this integral practically can be substitute with a finite sum over every position in the ranked sequence of documents:

$$\text{AveP} = \sum_{k=1}^{n} P(k)\Delta r(k)$$

In the equation above, k is the rank position in the sequence of retrieved documents, n is the total of retrieved documents, P(k) is the precision at cut-off k in the list, and Delta r(k) is the variance of recall from bin k-1 to bin k. Ten this discrete finite sum is equivalent to:

$$\text{AveP} = \frac{\sum_{k=1}^{n} P(k) \times \text{rel}(k)}{\text{number of relevant documents}}$$

And in the equation above, rel(k) is an indicator function which has value of 1 if the document ranked k is a relevant, 0 otherwise.

3.6.2 Mean average precision

Mean average precision for a set of queries is just defined as the mean of the average precision scores for each query. Thus, it can be calculated as:

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AveP(q)}}{Q}$$

where Q is the total number of queries.

Chapter 4

Experiment part1: Cross-media data pipeline

In our context of the project, there are at least two needs that should be fulfilled at the same time, since the project is solving both a practical engineering problem and a research problem of cross-media retrieval. Thus, during the procedure of the project, we should create a prototype of PDF books with video annotation and also assess the similarity of text and video contents in the meanwhile.

But actually, the two needs have been fulfilled at the same time as the project goes on. First, we collected 10 books in pdf format from different fields such as machine learning and programming language. Then we used PyPDF to extract the meta-data from the PDF books in order to get the outlines of each book with different hierarchies of titles. Next, we randomly chose 10 titles from each book as the queries that would be searched in YouTube. To simulate the search procedure in YouTube and collect the results, we used a tool provided by Google which is named Google custom search engine. Google custom search engine allows one to retrieve search results from google only with a pre-defined set of websites, which means that the behavior of search can be customized in some degree. Actually, in this step, we also tried to write some web crawler to extract contents of books that are in html format, but finally we gave up this method since it was hard to write a generalized web crawler which could be used with different html books.

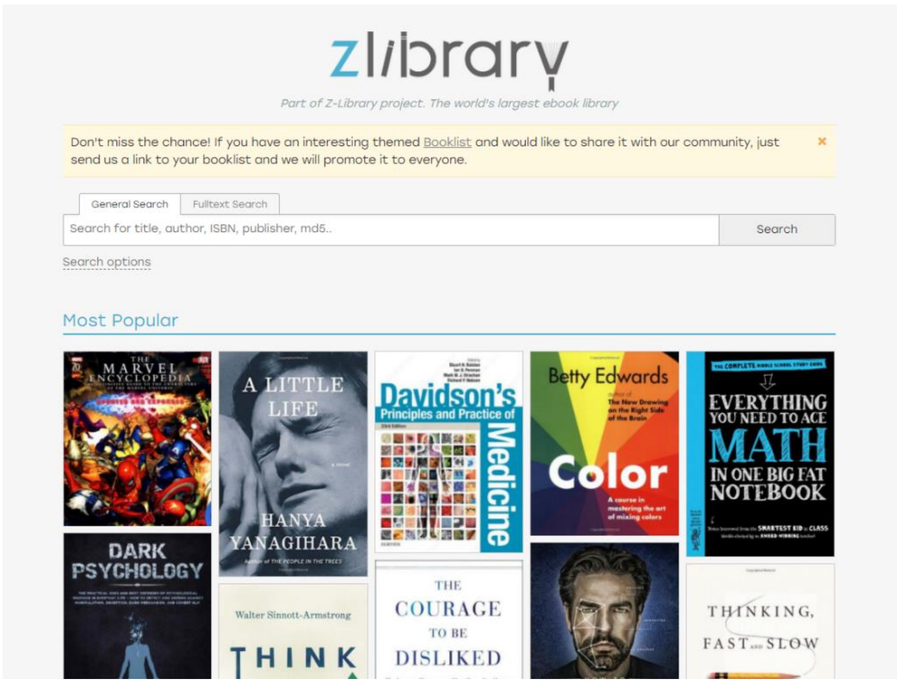
Moreover, a front-end presentation of the video annotated PDF book also has been implemented by using different techniques of front-end web development. More in details, the techniques used included html language, CSS language and JavaScript language. This front-end presentation can directly show the result of annotations of text content of a PDF book with most relevant YouTube videos, which may be really helpful for the later practical usage.

Finally, for the purpose of assessing the similarity between a pair of cross-media elements, here the text of a certain query and its relevant YouTube video, we used some NERD tools such as Babelfy and DBpedia spotlight to extract the named entities in the query and the subtitle of the video. Then we evaluated the similarity of the two sets of named entities and compared the results with the ground-truth.

And the first need can be seen as the construction of the input data pipeline of the second need, which means that the first need's output can be the input of the NLP based cross-media retrieval in the second part. Here in this chapter, let's talk about the procedure of the cross-media data pipeline construction.

4.1 Collect the books for the experiments

Before all the following steps, the first thing we should cope with is that we should collect a number of books to be the data resource of our experiments. At the beginning, we tried a lot to find websites which provided on-line books in PDF format or HTML format in good quality, since the books in PDF format should in a good enough quality for PyPDF. And finally, we found such a website which provided hundreds of e-books and we chose 10 of them to be the data resource of our next steps.


zlibrary
Part of Z-Library project. The world's largest ebook library
General Search
Fulltext Search
Search for title, author, ISBN, publisher, md5..
Search
Search options
Most Popular
THE MARVEL ENCYCLOPEDIA
A LITTLE LIFE
HANYA YANAGIHARA
Davidson's Principles and Practice of Medicine
Betty Edwards
Color
EVERYTHING YOU NEED TO ACE MATH IN ONE BIG FAT NOTEBOOK
DARK PSYCHOLOGY
Walter Sinnott-Armstrong
THINK
THE COURAGE TO BE DISLIKED
THINKING, FAST AND SLOW

Don't miss the chance! If you have an interesting themed Booklist and would like to share it with our community, just send us a link to your booklist and we will promote it to everyone.

Figure 4.1: The website used in which the 10 books in PDF format has been chosen from.

Then we chose e-books mainly from the domain computer science since this domain has abundant of learning resources on-line. And the books were from sub-domains of CS, such as machine learning, java programming language, Python programming language and etc. But even though we don't use the online books from other fields this time, I still believe that both the quantity and quality of the online books from all the other fields will also be increased rapidly in these years.

And here are the 10 selected books with PDF format in relatively high quality:

0_Introduction to algorith....org).pdf
1_Control System...ition.pdf
2_introduction to probabil...08).pdf
3_Semantic Web for the...b.org).pdf
4_Hand-on-ML.pdf
5_Java Program...org).pdf
6_Bishop.pdf
7_Practical SQL A Beginne....org).pdf
8_Python for Data Analysis...org).pdf
9_TensorFlow Machine...org).pdf

Figure 4.2: The 10 books collected as the data resource.

4.2 Text extraction from PDF books using pyPDF

After getting the 10 books collected in the last step, we used pyPDF which has been introduced in the last chapter to extract meta-data and text of the books. Since we wanted to make the titles as the queries of cross-media retrieval to retrieve relevant videos, we had to find a tool to extract titles in the table of contents of a book. Among the several choices such as tika and pyPDF, we finally decided to use pyPDF since it could be more flexible than the others.

For this purpose, several python scripts were created which encapsulate the python functions. First, we used pyPDF to extract the whole meta-data of a book. Then we got the outline of the book from the meta-data. And actually, there were 4 hierarchies of titles, which could be remarked as h1, h2, h3 and h4. After getting the outline, we randomly selected 10 titles of each book. Finally, we wrote another python script to clean the text of the chosen titles in order to make good quality queries.

Here is one of the 10 collected books, which has been regarded as the bible in machine learning area:

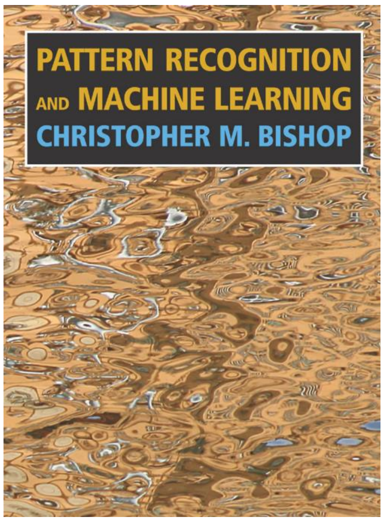


Figure 4.3: the book “pattern recognition and machine learning”.

And here is an example of h1 chapter titles extracted from this book:

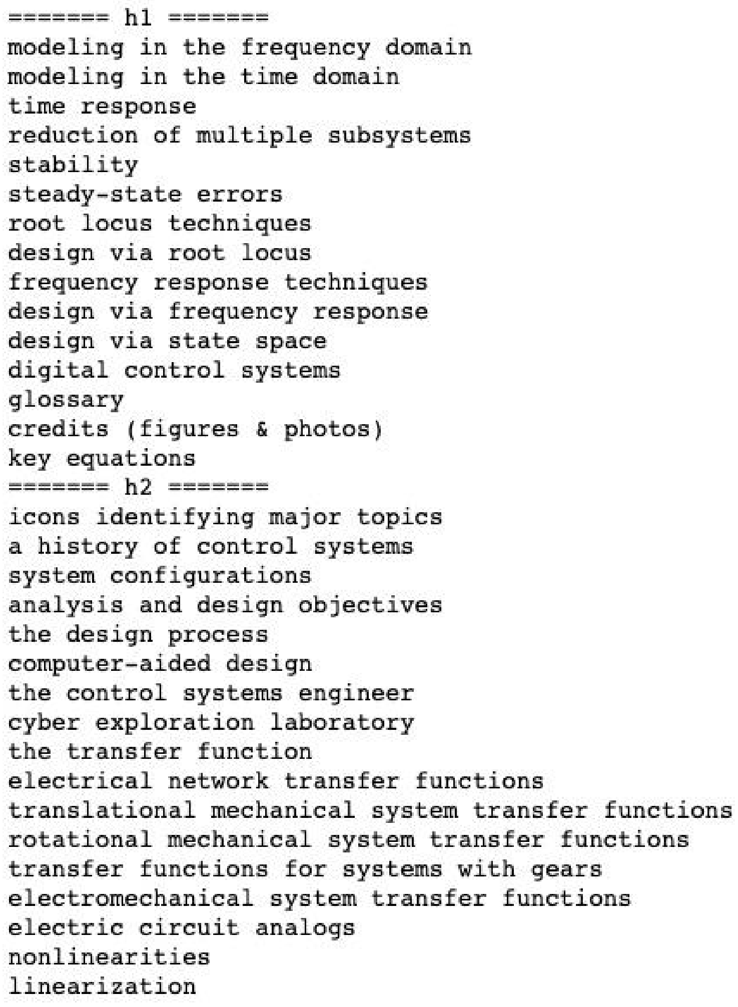
```
b' COVER\r'
Preface
Mathematical notation
Contents
b' 1. \rIntroduction'
2. Probability Distributions
3. Linear Models for Regression
4. Linear Models for Classification
5. Neural Networks
6. Kernel Methods
7. Sparse Kernel Machines
8. Graphical Models
9. Mixture Models and EM
10. Approximate Inference
11. Sampling Methods
12. Continuous Latent Variables
13. Sequential Data
14. Combining Models
Appendix A. Data Sets
Appendix B. Probability Distributions
Appendix C. Properties of Matrices
Appendix D. Calculus of Variations
Appendix E. Lagrange Multipliers
References
```

Figure 4.4: h1 chapters extracted from the book.

Actually, chapter titles of all the hierarchies (h1, h2, h3, h4) from the 10 collected books have been extracted and also the paragraphs related to each chapter title have been extracted and stored for later use. Moreover, since the raw texts extracted by pyPDF are not clean enough, thus the following steps should be done after the extraction by pyPDF:

1. Remove chapter titles with stop-words (like '\r' in windows system)
2. Delete symbols in chapter titles
3. Delete None-type items
4. Delete duplicated items

And here is an example of cleaned chapter titles:



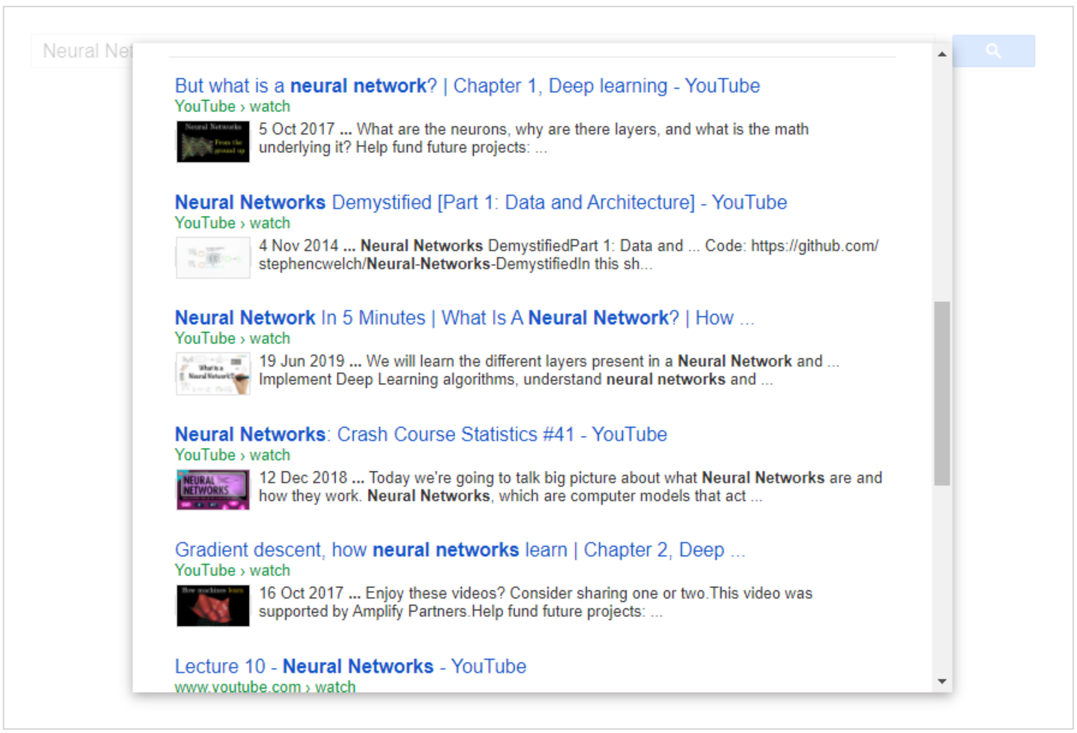
```
======= h1 =======
modeling in the frequency domain
modeling in the time domain
time response
reduction of multiple subsystems
stability
steady-state errors
root locus techniques
design via root locus
frequency response techniques
design via frequency response
design via state space
digital control systems
glossary
credits (figures & photos)
key equations
======= h2 =======
icons identifying major topics
a history of control systems
system configurations
analysis and design objectives
the design process
computer-aided design
the control systems engineer
cyber exploration laboratory
the transfer function
electrical network transfer functions
translational mechanical system transfer functions
rotational mechanical system transfer functions
transfer functions for systems with gears
electromechanical system transfer functions
electric circuit analogs
nonlinearities
linearization
```

Figure 4.5: an example of cleaned chapter titles.

4.3 Retrieve YouTube videos By google custom search

In this part, we used Google custom search engine to retrieve relevant video results according to the queries which were created in the last step. And it means that we searched in YouTube to find the most relevant videos in YouTube about a certain title in the book, which links two kinds of media together according to their specific semantics.

And here is an example of searching a query "Neural Networks" in a customized goolge search enigne:



Neural Ne

But what is a **neural network**? | Chapter 1, Deep learning - YouTube
YouTube › watch
5 Oct 2017 **...** What are the neurons, why are there layers, and what is the math underlying it? Help fund future projects: ...

Neural Networks Demystified [Part 1: Data and Architecture] - YouTube
YouTube › watch
4 Nov 2014 **...** **Neural Networks** DemystifiedPart 1: Data and ... Code: https://github.com/ stephencwelch/**Neural-Networks**-DemystifiedIn this sh...

Neural Network In 5 Minutes | What Is A **Neural Network**? | How ...
YouTube › watch
19 Jun 2019 **...** We will learn the different layers present in a **Neural Network** and ... Implement Deep Learning algorithms, understand **neural networks** and ...

Neural Networks: Crash Course Statistics #41 - YouTube
YouTube › watch
12 Dec 2018 **...** Today we're going to talk big picture about what **Neural Networks** are and how they work. **Neural Networks**, which are computer models that act ...

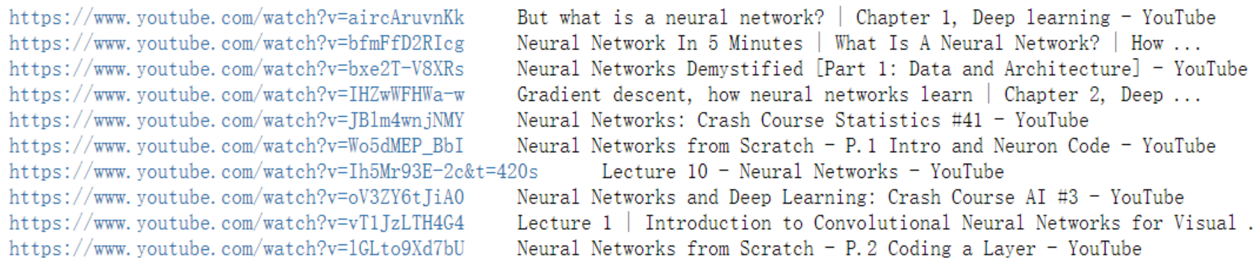
Gradient descent, how **neural networks** learn | Chapter 2, Deep ...
YouTube › watch
16 Oct 2017 **...** Enjoy these videos? Consider sharing one or two.This video was supported by Amplify Partners.Help fund future projects: ...

Lecture 10 - **Neural Networks** - YouTube
www.youtube.com › watch

Figure 4.6: result list of searching query "Neural Networks" in google custom search.

From the figure above, we can easily see that the results of the search is limited in the pre-defined set of websites (Here there is only one element in the set, the web site of YouTube). And that's the reason why google implemented such a project called "google custom searcg engine" which allow people to search in a self-defined set of websites not the whole Internet. Moreover, it also provides apis in different programming languages which allows people to use it in a programmable way in their own applications.

And here is an example of getting the search results list using the python api of google custom search using "Neural Networks" as query:



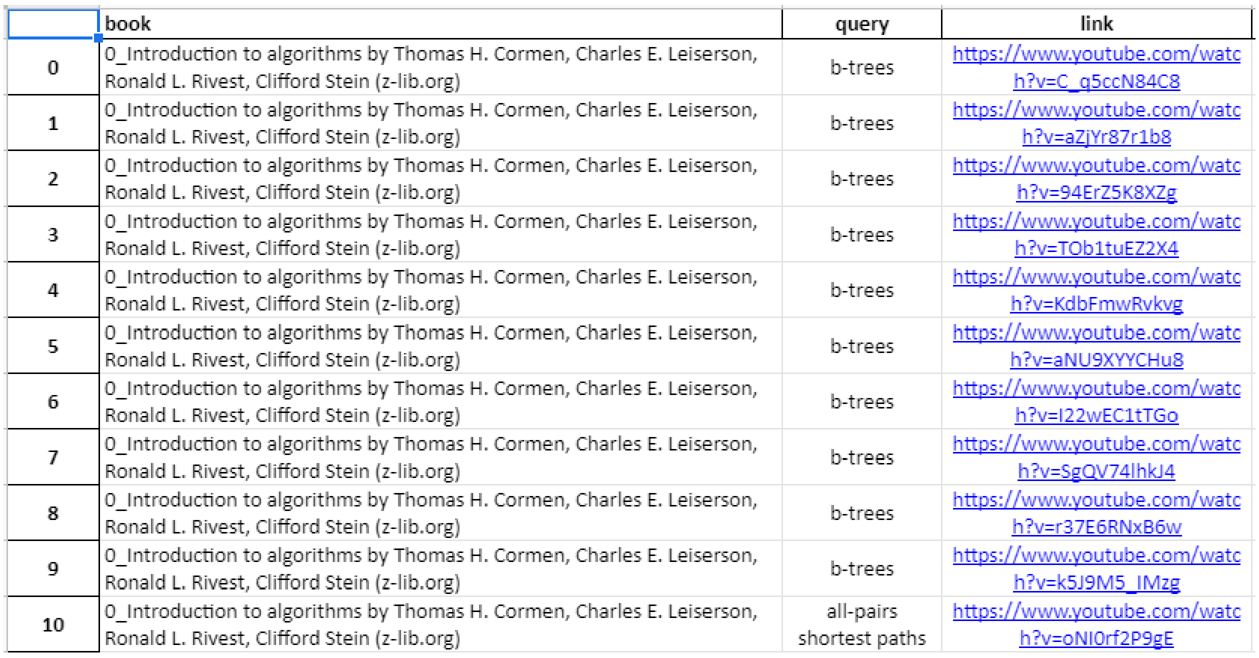
```
https://www.youtube.com/watch?v=aircAruvnKk      But what is a neural network? | Chapter 1, Deep learning - YouTube
https://www.youtube.com/watch?v=bfmFfD2RIcg      Neural Network In 5 Minutes | What Is A Neural Network? | How ...
https://www.youtube.com/watch?v=bxe2T-V8XRs      Neural Networks Demystified [Part 1: Data and Architecture] - YouTube
https://www.youtube.com/watch?v=IHZwWFHWa-w      Gradient descent, how neural networks learn | Chapter 2, Deep ...
https://www.youtube.com/watch?v=JB1m4wnjNMY      Neural Networks: Crash Course Statistics #41 - YouTube
https://www.youtube.com/watch?v=Wo5dMEP_BbI      Neural Networks from Scratch - P.1 Intro and Neuron Code - YouTube
https://www.youtube.com/watch?v=Ih5Mr93E-2c&t=420s      Lecture 10 - Neural Networks - YouTube
https://www.youtube.com/watch?v=oV3ZY6tJiA0      Neural Networks and Deep Learning: Crash Course AI #3 - YouTube
https://www.youtube.com/watch?v=vT1JzLTH4G4      Lecture 1 | Introduction to Convolutional Neural Networks for Visual .
https://www.youtube.com/watch?v=1GLto9Xd7bU      Neural Networks from Scratch - P.2 Coding a Layer - YouTube
```

Figure 4.7: returned links and titles of the YouTube videos searching "Neural Networks" in a python program.

From the figure above, we can see that we limit the number of the returned video results for each query as 10. Since in this work, we only focus on the ranking and the quality of the top 10 results of the search. And then the top 10 YouTube video results are stored as the candidate lists of the final result. What's more, for each of the collected book, we randomly choose 10 of the extracted chapter titles as queries.

Thus, there are totally 10 books collected as the data resources, and there are 10 randomly selected chapter titles of each book to be used as queries, and then the top 10 video results returned by YouTube using google custom search for each query are stored. As a result, the total dataset consists of 100 of queries and 1000 pairs of text chapter title and YouTube video. And later in the next chapter, this dataset will be labeled manually and utilized as the whole dataset for the training and validation of machine learning models.

And here is part of the collected dataset:



	book	query	link
0	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=C_q5ccN84C8
1	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=aZjYr87r1b8
2	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=94ErZ5K8XZg
3	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=TOb1tuEZ2X4
4	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=KdbFmwRvkvg
5	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=aNU9XYYCHu8
6	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=I22wEC1tTGo
7	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=SgQV74lhkJ4
8	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=r37E6RNxB6w
9	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=k5J9M5_IMzg
10	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=oNI0rf2P9gE

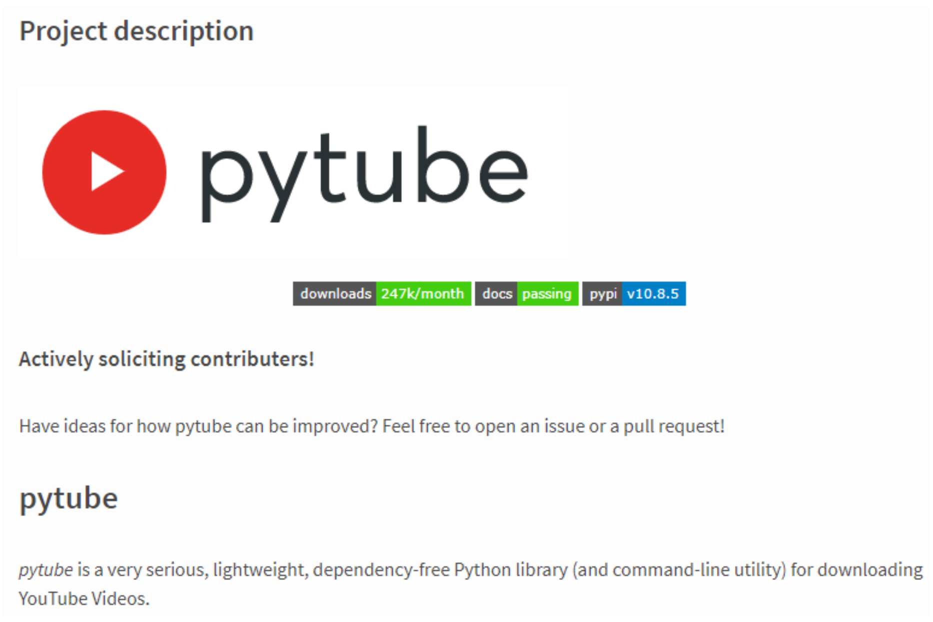
Figure 4.8: part of the collected dataset which consists of pairs between text queries and returned candidate YouTube videos.

From the figure above, we can see that there are mainly three different columns in the dataset. The first one is the name of the book which is the source of the query chapter title. Then the second one is the query (randomly selected from the book with the book name), which is used to search in google custom search engine. And the last one is the link of the YouTube video result, which will be assessed whether is matched with the text query or not.

But there is only a problem of using the free version of google custom search engine, since there is a limitation of search number as 1000 each day, thus if there are some more flexible substituttes, they should be also considered to replace google custom search engine.

4.4 YouTube video captions Extraction by pyTube

After retrieving candidate related YouTube videos for each query, we regard the subtitles of each video contains most parts of the natural language information of that video, which represents the main semantic meanings of the video contents. Since we target to retrieve educational YouTube videos in this work, the famous python package called pyTube should be considered first.



Project description

pytube

downloads 247k/month docs passing pypi v10.8.5

Actively soliciting contributers!

Have ideas for how pytube can be improved? Feel free to open an issue or a pull request!

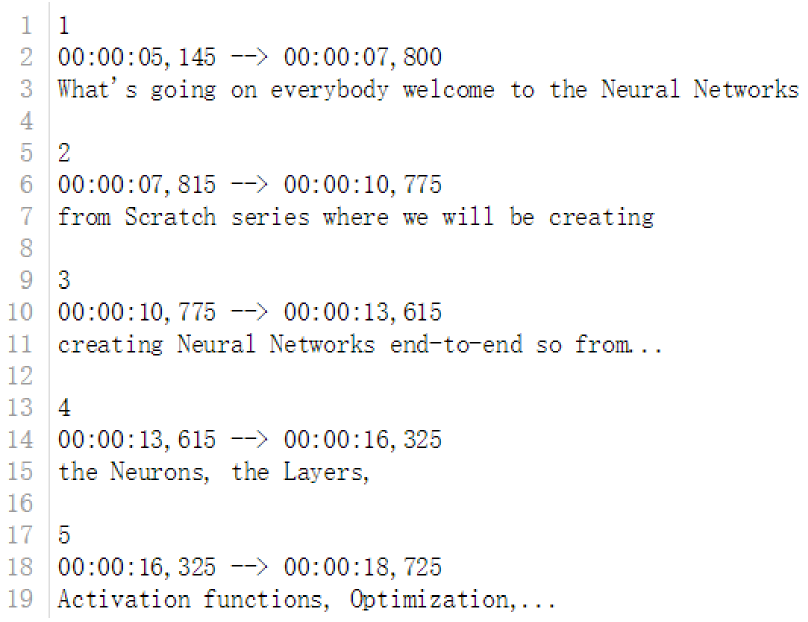
pytube

pytube is a very serious, lightweight, dependency-free Python library (and command-line utility) for downloading YouTube Videos.

Figure 4.9: the home page of pyTube package.

pyTube is very capable of obtaining nearly or the information contained in a YouTube video. Not only it can be used to download YouTube videos in a python program, it can also be used to retrieve video titles, captions and all the other meta-data of a YouTube video. As a result, it's a fantastic choice whenever you want to develop an application related to YouTube videos.

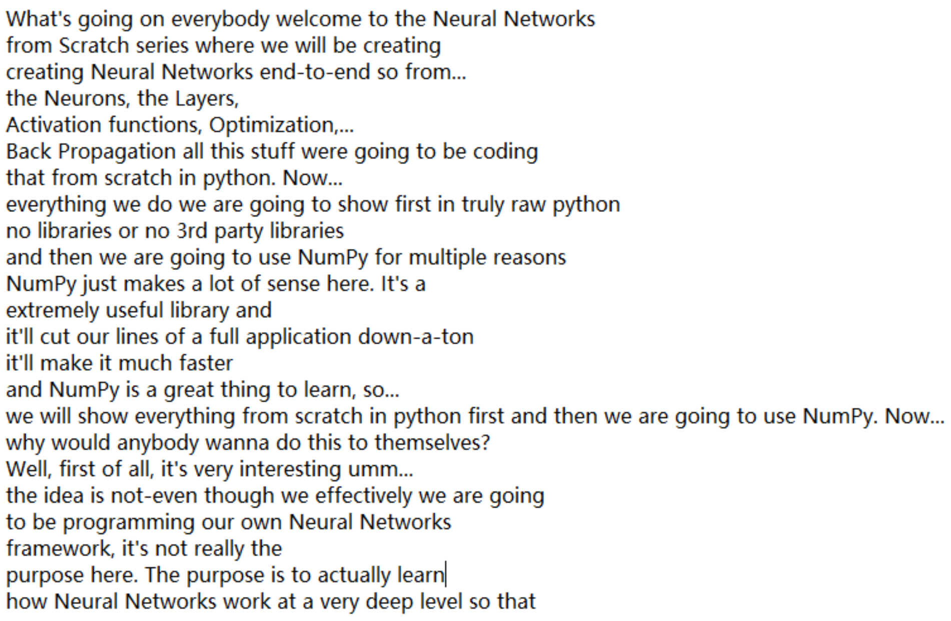
And here is an example of the caption extraction of the YouTube video: https://www.youtube.com/watch?v=Wo5dMEP_BbI, and the raw caption should be converted to a format of caption called srt for easily reading:



```
1
00:00:05,145 --> 00:00:07,800
What's going on everybody welcome to the Neural Networks

2
00:00:07,815 --> 00:00:10,775
from Scratch series where we will be creating

3
00:00:10,775 --> 00:00:13,615
creating Neural Networks end-to-end so from...

4
00:00:13,615 --> 00:00:16,325
the Neurons, the Layers,

5
00:00:16,325 --> 00:00:18,725
Activation functions, Optimization,...
```

Figure 4.10: an example of the retrieved caption is srt format of a YouTube video.

And after obtaining the srt format of a specific caption, we can easily extract the text of the speak and save them in another file. And here is an example of saved file with speak text of a video caption:



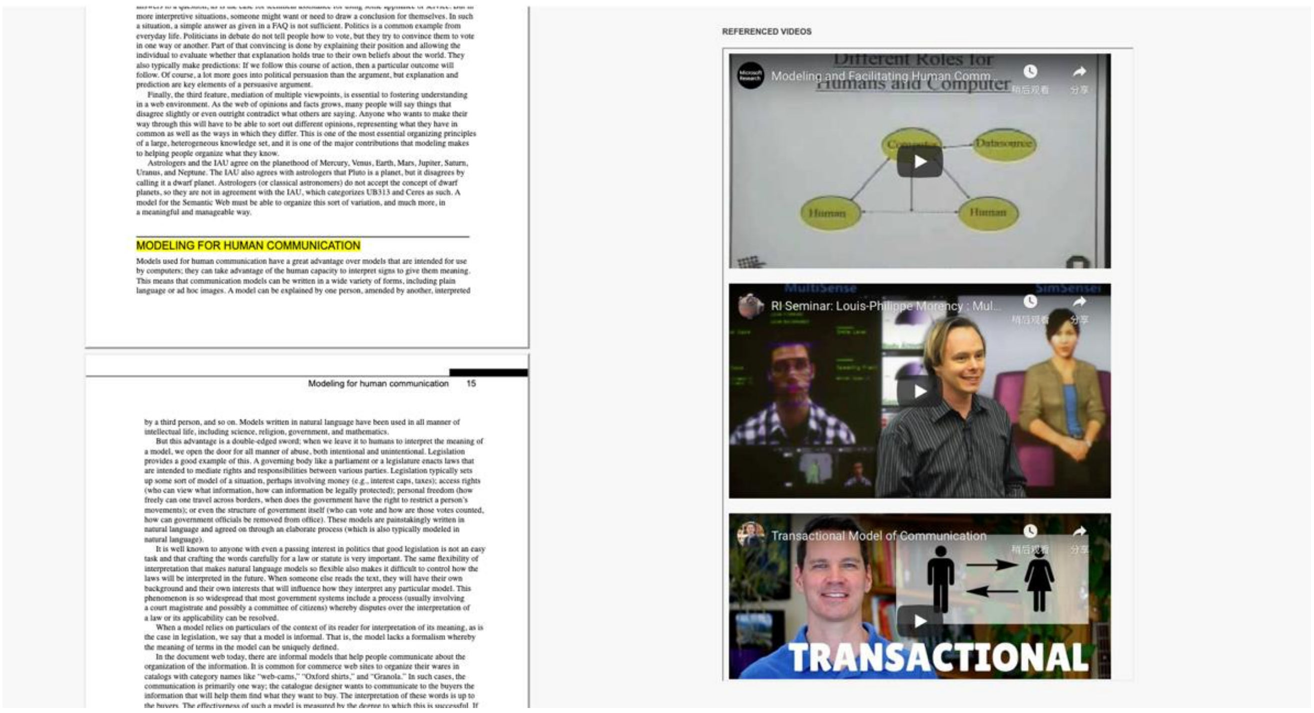
```
What's going on everybody welcome to the Neural Networks
from Scratch series where we will be creating
creating Neural Networks end-to-end so from...
the Neurons, the Layers,
Activation functions, Optimization,...
Back Propagation all this stuff were going to be coding
that from scratch in python. Now...
everything we do we are going to show first in truly raw python
no libraries or no 3rd party libraries
and then we are going to use NumPy for multiple reasons
NumPy just makes a lot of sense here. It's a
extremely useful library and
it'll cut our lines of a full application down-a-ton
it'll make it much faster
and NumPy is a great thing to learn, so...
we will show everything from scratch in python first and then we are going to use NumPy. Now...
why would anybody wanna do this to themselves?
Well, first of all, it's very interesting umm...
the idea is not-even though we effectively we are going
to be programming our own Neural Networks
framework, it's not really the
purpose here. The purpose is to actually learn
how Neural Networks work at a very deep level so that
```

Figure 4.11: an example of the pure text of YouTube video caption.

4.5 A presentation created by front-end languages

In this thesis, a prototype of visualization or presentation of the final result has been implemented also. Since the whole work can be seen as a prototype of a application which provides the students or learners a annotation system to annotate chapter titles with relevant YouTube videos, we want to make it visually touchable directly for the future users that it really can help a lot for the efficiency of study.

And here is a figure of the result product which presents the original PDF book on the left, and the YouTube video list related to a yellow-highlighted chapter title on the right:



more interpretive situations, someone might want or need to draw a conclusion for themselves. In such a situation, a simple answer as given in a FAQ is not sufficient. Politics is a common example from everyday life. Politicians in debate do not tell people how to vote, but they try to convince them to vote in one way or another. Part of that convincing is done by explaining their position and allowing the individual to evaluate whether that explanation holds true to their own beliefs about the world. They also typically make predictions: If we follow this course of action, then a particular outcome will follow. Of course, a lot more goes into political persuasion than the argument, but explanation and prediction are key elements of a persuasive argument.

Finally, the third feature, mediation of multiple viewpoints, is essential to fostering understanding in a web environment. As the web of opinions and facts grows, many people will say things that disagree slightly or even outright contradict what others are saying. Anyone who wants to make their way through this will have to be able to sort out different opinions, representing what they have in common as well as the ways in which they differ. This is one of the most essential organizing principles of a large, heterogeneous knowledge set, and it is one of the major contributions that modeling makes to helping people organize what they know.

Astrologers and the IAU agree on the planethood of Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, and Neptune. The IAU also agrees with astrologers that Pluto is a planet, but it disagrees by calling it a dwarf planet. Astrologers (or classical astronomers) do not accept the concept of dwarf planets, so they are not in agreement with the IAU, which categorizes UB313 and Ceres as such. A model for the Semantic Web must be able to organize this sort of variation, and much more, in a meaningful and manageable way.

MODELING FOR HUMAN COMMUNICATION

Models used for human communication have a great advantage over models that are intended for use by computers; they can take advantage of the human capacity to interpret signs to give them meaning. This means that communication models can be written in a wide variety of forms, including plain language or ad hoc images. A model can be explained by one person, amended by another, interpreted

Modeling for human communication 15

by a third person, and so on. Models written in natural language have been used in all manner of intellectual life, including science, religion, government, and mathematics.

But this advantage is a double-edged sword; when we leave it to humans to interpret the meaning of a model, we open the door for all manner of abuse, both intentional and unintentional. Legislation provides a good example of this. A governing body like a parliament or a legislature enacts laws that are intended to mediate rights and responsibilities between various parties. Legislation typically sets up some sort of model of a situation, perhaps involving money (e.g., interest caps, taxes); access rights (who can view what information, how can information be legally protected); personal freedom (how freely can one travel across borders, when does the government have the right to restrict a person's movements); or even the structure of government itself (who can vote and how are those votes counted, how can government officials be removed from office). These models are painstakingly written in natural language and agreed on through an elaborate process (which is also typically modeled in natural language).

It is well known to anyone with even a passing interest in politics that good legislation is not an easy task and that crafting the words carefully for a law or statute is very important. The same flexibility of interpretation that makes natural language models so flexible also makes it difficult to control how the laws will be interpreted in the future. When someone else reads the text, they will have their own background and their own interests that will influence how they interpret any particular model. This phenomenon is so widespread that most government systems include a process (usually involving a court magistrate and possibly a committee of citizens) whereby disputes over the interpretation of a law or its applicability can be resolved.

When a model relies on particulars of the context of its reader for interpretation of its meaning, as is the case in legislation, we say that a model is informal. That is, the model lacks a formalism whereby the meaning of terms in the model can be uniquely defined.

In the document web today, there are informal models that help people communicate about the organization of the information. It is common for commerce web sites to organize their wares in catalogs with category names like "web-cams," "Oxford shirts," and "Granola." In such cases, the communication is primarily one way; the catalogue designer wants to communicate to the buyers the information that will help them find what they want to buy. The interpretation of these words is up to the buyers. The effectiveness of such a model is measured by the degree to which this is successful. If

REFERENCED VIDEOS
Modeling and Facilitating Human Comm
Different Roles for Humans and Computer
Computer
Datasource
Human
Human
RI Seminar: Louis-Philippe Morency : Mul...
MultiSense
SimSensei
Transactional Model of Communication
TRANSACTIONAL

Figure 4.11: a visual presentation of the final result, original PDF book on the left, related YouTube video list of a chapter title on the right.

From the figure above, we can see that the current presentation is still very simple, and it can be improved a lot by more complicated usage of the front-end languages such as HTML, CSS and JavaScript. And in order to implement this presentation, HTML is used to define the elements in a web page, and CSS is used to layout the defined elements properly, then JavaScript make it possible to interact between the dynamic web page (since the reader should be able to go up and down the PDF book) and the readers.

Chapter 5
Experiment 2: NLP-based Cross-media retrieval and result analysis

Finally, in order to improve the quality of the search result of cross-media retrieval.
We labeled one thousand pairs of text queries and searched video results as samples, and then trained machine learning models to classify among the positive samples (labeled as 1 since we regard the pair of cross-media information are matched) and negative ones (labeled as 0 since we regard the pair of cross-media information are not matched enough). Then we create 10 different datasets for each book and train 10 different Xgboost classifiers. And after getting the trained machine learning model, we use it to predict test set of samples which not participate the training of the model. Next, all the predicted negative ones are filtered out and we compare the metric called mean average precision before and after the filter to see whether such kind of classification by the trained model can improve the quality of cross-media retrieval.

As a result, the experiments shows that the mean average precision @k (k=1, 2, ..., 10) all increase by around 10% after filter out the predicted negative candidate YouTube videos for the queries, which means that the quality of the search result has been improved obviously.

And along this journey of machine learning model training, there are many different difficulties that we should face to. And the biggest one is the lack of labeled data, since the manually check whether a pair of text query and returned video can be really time-cost. Thus, under the condition in which there are only 1000 samples totally, we still should find the best and reasonable way to train machine learning models and find out the best ones.

And the experiment of the above process can be split as four different parts: 1. The preparation of dataset. 2. The extraction of features to make the samples for the machine learning models. 3. The training of the machine learning model. 4. The calculation of the mean average precision @ k (k=1, 2, ..., 10) and result analysis.

5.1 Preparation of the dataset

Since the underlying application of our research is to provide a function that annotates the important keywords in a book with relevant YouTube videos. Thus, our raw data are from text of books and videos of YouTube. And since we regard the text (especially the titles of chapters) as the search query and YouTube videos as the search results, we should first select several different books as the source of the query. Here are the 10 books we selected which have already been introduced previously:

0_Introduction to algorith....org).pdf
1_Control System...ition.pdf
2_introduction to probabil...08).pdf
3_Semantic Web for the...b.org).pdf
4_Hand-on-ML.pdf
5_Java Program...org).pdf
6_Bishop.pdf
7_Practical SQL A Beginne....org).pdf
8_Python for Data Analysis...org).pdf
9_TensorFlow Machine...org).pdf

Figure 5.1: The 10 selected books as data resource.

As introduced in the previous chapter, after getting all the text using PyPDF, 10 chapter titles have been randomly selected from each of the 10 books as the queries. Thus, there are totally 100 queries. And for each of the query, only the top 10 retrieved YouTube videos have been kept as the candidate video results for each query. As a result, there are totally 1000 pairs of text queries and video results in the constructed dataset.

And there are some problems of the labeling standard during the procedure of labeling the dataset, since there are no general standard criteria to judge whether a video is related to a text query or not. And the criteria can be really subjective. Thus, several discussions about the labeling criteria have been done with the colleges, and several different criteria have been ever utilized during the process. Here two of the different criteria will be introduced:

The first criteria can be descripted as:

1. Absolutely matched and the video helps a lot understanding the content of the text chapter title will be labeled as '1'.
2. Almost matched and the video helps in some degree understanding the content of the text chapter title will be labeled as '0.75'.
3. Basically matched video will be labeled as '0.5'.
4. A little bit matched video will be labeled as '0.25'.
5. Absolutely not matched video will be labeled as '0'.

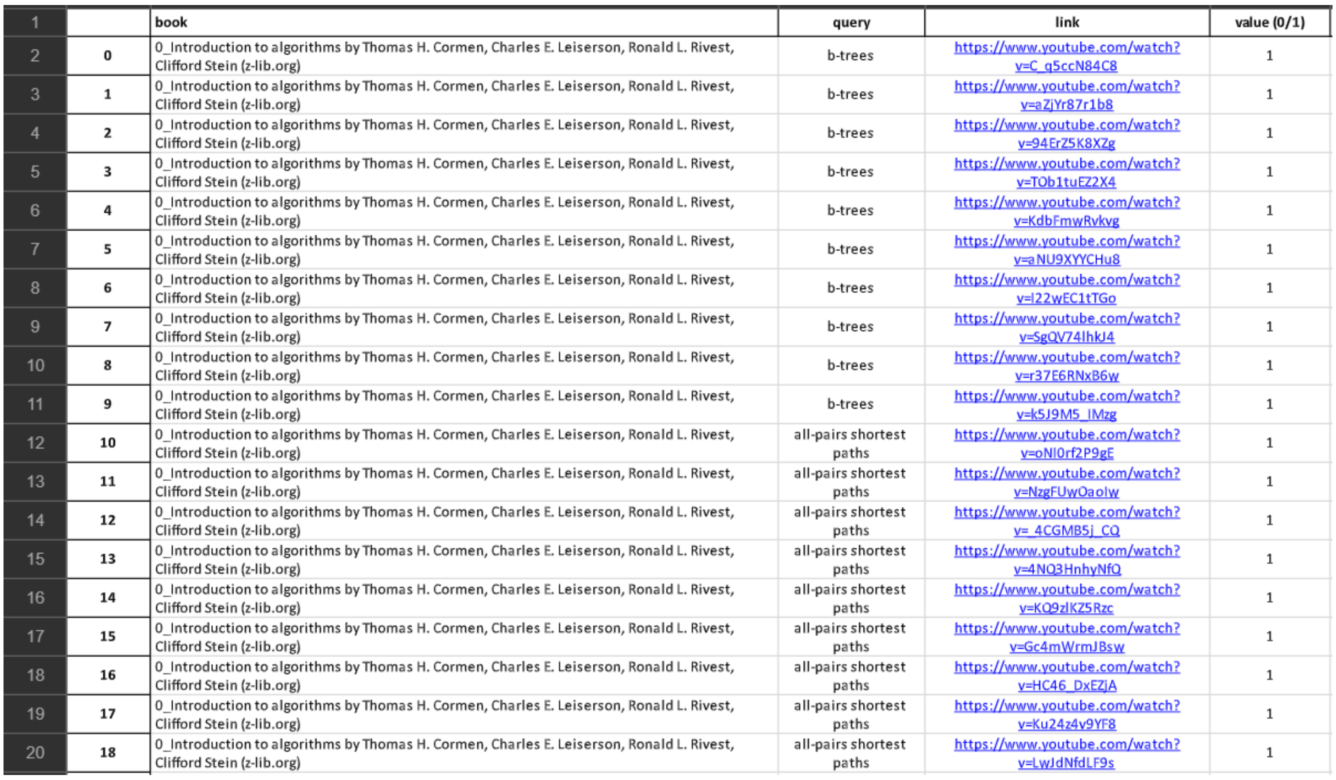
And the second criteria can be descripted as:

1. Absolutely matched and the video helps a lot understanding the content of the text chapter title will be labeled as '1'.
2. Not absolutely matched and the video can't be regarded as really helpful for understanding the according text content will be labeled as '0'.

Compare the above two criteria, there is a trade-off between complexity and simplicity. Since although the first criteria seems to be much more concrete and reasonable, but under the condition that we only have 1000 samples to be used. If we split the 1000 samples into five classes as descripted in the first criteria, the number of each class will be very limited. Moreover, it will be also much harder for us to label the data into five different classes since sometimes we are not that confirmed about the fine-grained degree of match between the result video and text query.

As a result, the second criteria descripted above has finally been chosen to be utilized according to the consider of both the limit of the size of the dataset and the simplicity of labeling work.

And here is part of the labeled data which follows the chosen criteria:



1		book	query	link	value (0/1)
2	0	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=C_q5ccN84C8	1
3	1	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=aZjYr87r1b8	1
4	2	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=94ErZ5K8XZg	1
5	3	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=TOb1tuEZ2X4	1
6	4	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=KdbFmwRvkvg	1
7	5	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=aNU9XYYCHu8	1
8	6	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=I22wEC1tTGo	1
9	7	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=SgQV74lhkJ4	1
10	8	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=r37E6RNxB6w	1
11	9	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	b-trees	https://www.youtube.com/watch?v=k5J9M5_IMzg	1
12	10	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=oNI0rf2P9gE	1
13	11	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=NzgFUwOaolw	1
14	12	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=_4CGMB5j_CQ	1
15	13	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=4NQ3HnhyNfQ	1
16	14	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=KQ9zlKZ5Rzc	1
17	15	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=Gc4mWrmJBsw	1
18	16	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=HC46_DxEZjA	1
19	17	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=Ku24z4v9YF8	1
20	18	0_Introduction to algorithms by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (z-lib.org)	all-pairs shortest paths	https://www.youtube.com/watch?v=LwJdNfdLF9s	1

Figure 5.2: part of the labeled dataset according to the second criteria.

And there is still an important consider should be coped with is that we can not directly utilize the whole obtained dataset and split it into training set and test set, since our target is to improve the result ranking list for each query, thus the whole dataset can be seen as a set of different groups of subsets, and in each subsets there are 10 samples which belong to the same query and composites the whole result ranking list of the according query. In addition, some of the result videos don't have a manually or automatically created subtitle which can't be utilized for the later part of feature engineering. As a result, only 790 samples left to be used as the whole dataset.

According to the previously descripted difficulty of lacking of data, a clever way should be found to create different datasets for training models to filter out not matched videos for each query.

As a result, we decide to create 10 different datasets and train 10 different models separately. And the criteria of the dataset creation can be descripted as following:

1. Use all the samples of the queries belong to the same book as the test set.
2. Use all the other samples of the queries belong to all the other books as the training set.

And the reason why we create the datasets according to the criteria descripted above is very trivial, and there are mainly two considers:

1. The scale of each training set should be as big as possible, and the scale of each test set also shouldn't be too limited.
2. Whenever we want to assess the result of the trained model on a specific test set, the training set utilized to train the model should be orthogonal to the test set, which means the training set and the test set should be absolutely independent on each other.

Thus, we create 10 different datasets according to the criteria and the statistics of the 10 created datasets are shown in the figures below.

Here is the figure about the label distribution of the two classes (positive and negative) of the 10 training sets:

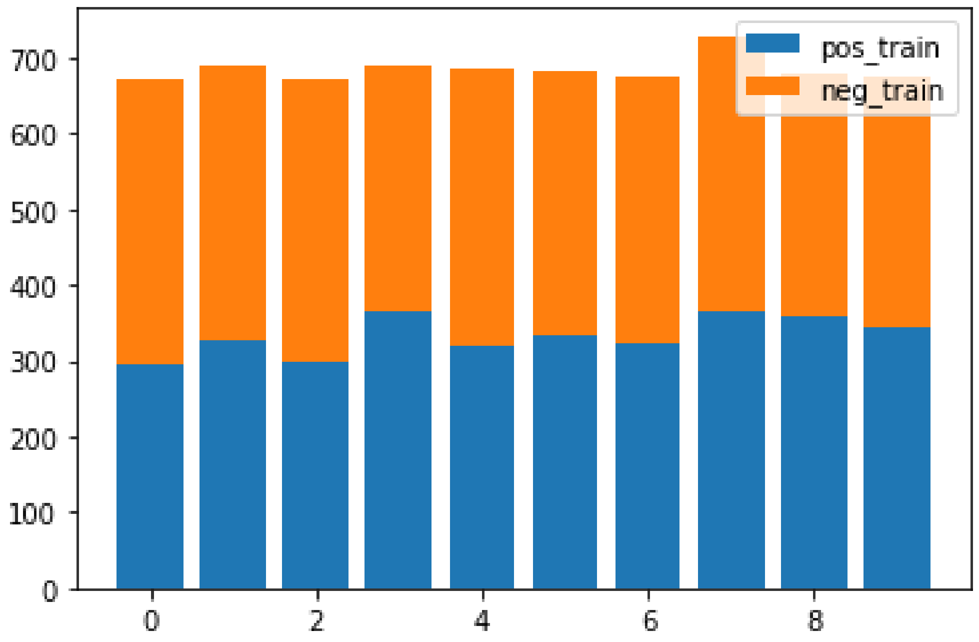


Figure 5.3: label distribution of the two classes (positive and negative) of the 10 created training sets.

From the figure above, we can see that the distributions of the two classes are really balanced in all the 10 training sets, which mean the number of the two classes are almost equal in each set.

And here is the figure about the label distribution of the two classes (positive and negative) of the 10 test sets:

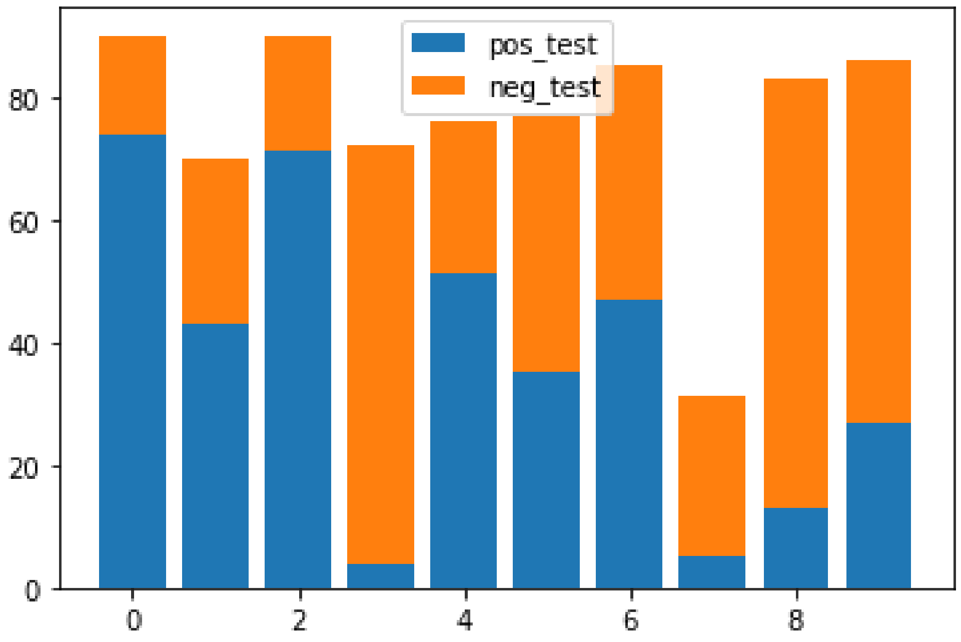


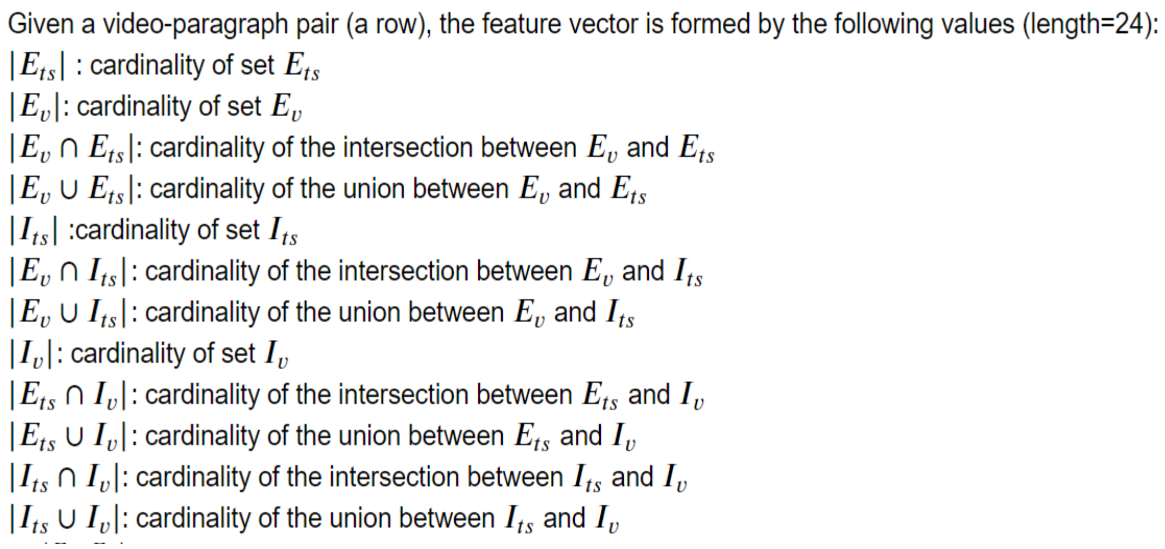
Figure 5.3: label distribution of the two classes (positive and negative) of the 10 created test sets.

From the figure above, we can see that the distributions of the two class (positive and negative) of the 10 test sets are quite different from their relevant training sets and quite imbalanced, which will lead to large differences of the model assessments on the 10 different test sets from two reasons:

1. Although the training sets are all balanced and consistent among the 10 training sets, the quite various label distributions of the test sets will have quite different assessment metric values, such as precision, recall, f1-score and AUC.
2. Still the problem of lack of data, we can see that the sizes of all the test sets are under 100.

5.2 Feature engineering

In order to create the feature vectors of each sample as the input of machine learning models or classifiers, we need to design the features that will be used. Since our target is to measure the similarity between the text of a chapter in book and the text of the subtitle of a video. And according to the similarities descripted in [29], [30], we take advantage of similarities between set of named entities and set of instance entities of each resource and used several different kinds of derived set similarities as different features. As a result, we designed and got 24 different features for each sample as shown below:



Given a video-paragraph pair (a row), the feature vector is formed by the following values (length=24):

$|E_{ts}|$: cardinality of set E_{ts}

$|E_v|$: cardinality of set E_v

$|E_v \cap E_{ts}|$: cardinality of the intersection between E_v and E_{ts}

$|E_v \cup E_{ts}|$: cardinality of the union between E_v and E_{ts}

$|I_{ts}|$:cardinality of set I_{ts}

$|E_v \cap I_{ts}|$: cardinality of the intersection between E_v and I_{ts}

$|E_v \cup I_{ts}|$: cardinality of the union between E_v and I_{ts}

$|I_v|$: cardinality of set I_v

$|E_{ts} \cap I_v|$: cardinality of the intersection between E_{ts} and I_v

$|E_{ts} \cup I_v|$: cardinality of the union between E_{ts} and I_v

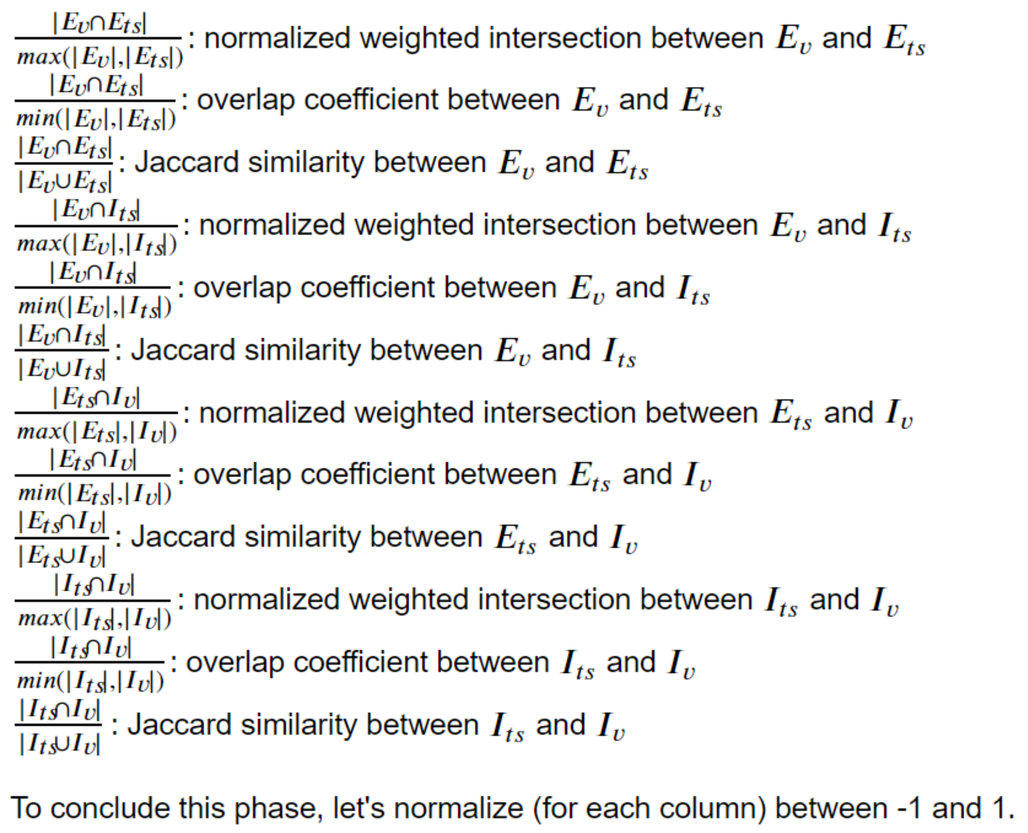
$|I_{ts} \cap I_v|$: cardinality of the intersection between I_{ts} and I_v

$|I_{ts} \cup I_v|$: cardinality of the union between I_{ts} and I_v

Figure 5.4: first 12 features utilized as the input of machine learning models.

Here a brief explanation of the presented names will be done:

1. E_{ts} is the set of named entities in a text snippet.
2. E_v is the set of named entities in a video.
3. I_{ts} is the set of instance entities in a text snippet which should be a subset of E_{ts}.
4. I_v is the set of instance entities in a video which should be a subset of E_v.



$\frac{|E_v \cap E_{ts}|}{max(|E_v|,|E_{ts}|)}$: normalized weighted intersection between E_v and E_{ts}

$\frac{|E_v \cap E_{ts}|}{min(|E_v|,|E_{ts}|)}$: overlap coefficient between E_v and E_{ts}

$\frac{|E_v \cap E_{ts}|}{|E_v \cup E_{ts}|}$: Jaccard similarity between E_v and E_{ts}

$\frac{|E_v \cap I_{ts}|}{max(|E_v|,|I_{ts}|)}$: normalized weighted intersection between E_v and I_{ts}

$\frac{|E_v \cap I_{ts}|}{min(|E_v|,|I_{ts}|)}$: overlap coefficient between E_v and I_{ts}

$\frac{|E_v \cap I_{ts}|}{|E_v \cup I_{ts}|}$: Jaccard similarity between E_v and I_{ts}

$\frac{|E_{ts} \cap I_v|}{max(|E_{ts}|,|I_v|)}$: normalized weighted intersection between E_{ts} and I_v

$\frac{|E_{ts} \cap I_v|}{min(|E_{ts}|,|I_v|)}$: overlap coefficient between E_{ts} and I_v

$\frac{|E_{ts} \cap I_v|}{|E_{ts} \cup I_v|}$: Jaccard similarity between E_{ts} and I_v

$\frac{|I_{ts} \cap I_v|}{max(|I_{ts}|,|I_v|)}$: normalized weighted intersection between I_{ts} and I_v

$\frac{|I_{ts} \cap I_v|}{min(|I_{ts}|,|I_v|)}$: overlap coefficient between I_{ts} and I_v

$\frac{|I_{ts} \cap I_v|}{|I_{ts} \cup I_v|}$: Jaccard similarity between I_{ts} and I_v

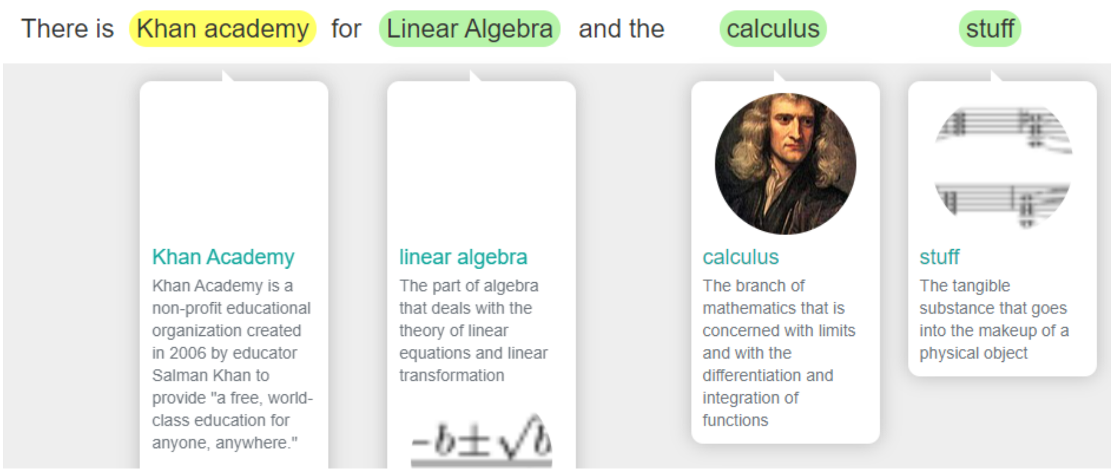
To conclude this phase, let's normalize (for each column) between -1 and 1.

Figure 5.5: last 12 features utilized as the input of machine learning models.

Here we will briefly introduce what is an instance named entity. Usually the named entities should be collected and stored in a specific knowledge base such as wikidata, DBpedia or Yago. And among the different knowledge bases, wikidata is the most popular one. Moreover, there are several different knowledge base tools such as TextRazor or Babelfy which are based on wikidata. In addition, the named entities in wikidata can be regarded as two different classes:

1. Class named entity: which is the abstract name of a class of objects, such as human being, animal, and city, which are not referenced to a specific concrete exist.
2. Instance named entity: which are referenced to a specific concrete exists, such as Cristiano Ronaldo, Leaning Tower of Pisa. And they always belong to some abstract classes.

Thus, in order to implement the features descripted above, we should collect both the sets of named entities and instance entities in the pair of text snippets and videos separately. In this thesis, Babelfy introduced in chapter 3 has been utilized to extract and link the named entities in the text, and here is an example of the named entity linking from a sentence in the subtitle of a video:



There is Khan academy for Linear Algebra and the calculus stuff

Khan Academy
Khan Academy is a non-profit educational organization created in 2006 by educator Salman Khan to provide "a free, world-class education for anyone, anywhere."

linear algebra
The part of algebra that deals with the theory of linear equations and linear transformation

calculus
The branch of mathematics that is concerned with limits and with the differentiation and integration of functions

stuff
The tangible substance that goes into the makeup of a physical object

Figure 5.6: a presentation of NEL by Babelfy.

In practice, we invoke the NEL api provided by Babelfy to link named entities in text from wikidata in python programs. And now we can obtain the set of named entities (E_{ts} *and* E_v) of each pair of cross-media resources, then how can we obtain the instance entity sets (I_{ts} *and* I_v) of each pair of cross-media resources?

First we try to check out all the instances in E_{ts} or E_v by querying the Wikidata Sparql endpoint using the following Sparql sentence:

```
ASK {
  <ENTITY URI> wdt:P31 ?o.
}
```

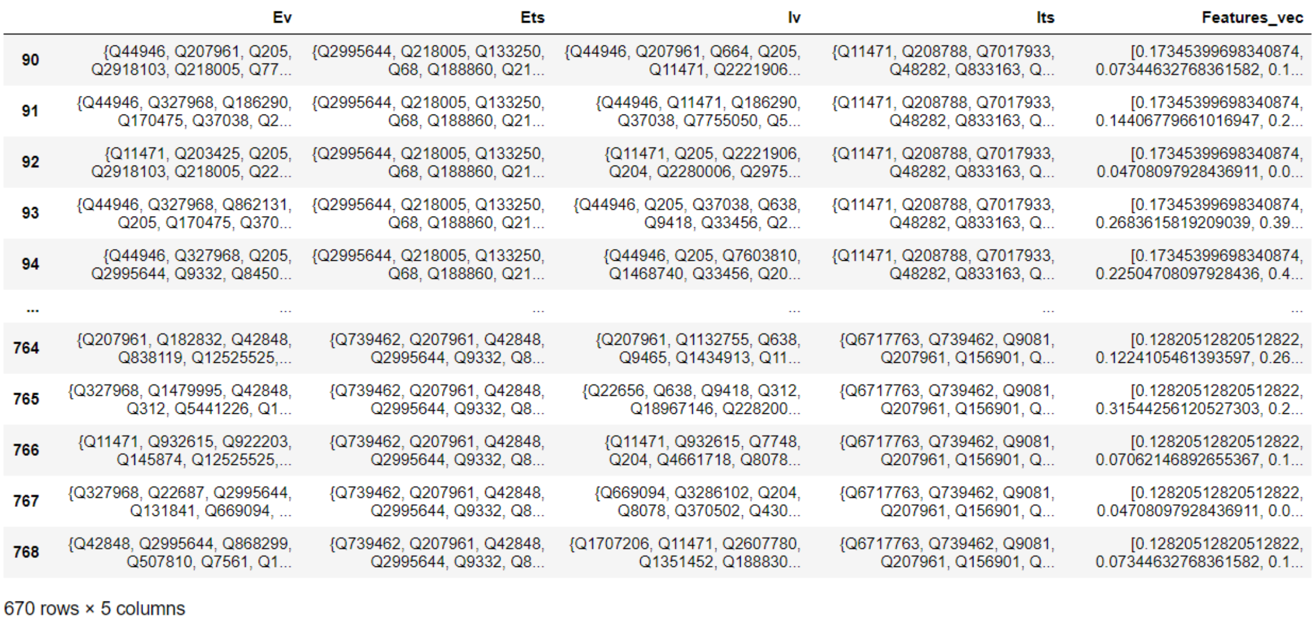
Figure 5.7: Sparql sentence to distinguish an instance named entity.

In the Sparql sentence above, <ENTITY URI> indicates the specific URI of the named entity that has been queried. And wdt:P31 is a pre-defined predicate in Wikidata knowledge base which means a named entity is an instance of any abstract class.

But for some unknown reason, the returned results of all the queries are all false, it means none of the checked named entities is an instance named entity, which is absolutely wrong. Then we turn to use the python library called requests to directly access to the online saved json object which consists of all the Wikidata named entities and check whether the predicate P31 is one of their attributes.

As a result, until now we are able to obtain E_{ts} , E_v, I_{ts} *and* I_v which are needed to construct the designed features for a pair of cross-media resources, and after a series of calculations, the feature vectors can be obtained and utilized as inputs to machine learning models.

And here is the presentation of part of the new datasets with E_{ts} , E_v , I_{ts} *and* I_v, and also the calculated feature vectors:



	Ev	Ets	Iv	Its	Features_vec
90	{Q44946, Q207961, Q205, Q2918103, Q218005, Q77...	{Q2995644, Q218005, Q133250, Q68, Q188860, Q21...	{Q44946, Q207961, Q664, Q205, Q11471, Q2221906...	{Q11471, Q208788, Q7017933, Q48282, Q833163, Q...	[0.17345399698340874, 0.07344632768361582, 0.1...
91	{Q44946, Q327968, Q186290, Q170475, Q37038, Q2...	{Q2995644, Q218005, Q133250, Q68, Q188860, Q21...	{Q44946, Q11471, Q186290, Q37038, Q7755050, Q5...	{Q11471, Q208788, Q7017933, Q48282, Q833163, Q...	[0.17345399698340874, 0.14406779661016947, 0.2...
92	{Q11471, Q203425, Q205, Q2918103, Q218005, Q22...	{Q2995644, Q218005, Q133250, Q68, Q188860, Q21...	{Q11471, Q205, Q2221906, Q204, Q2280006, Q2975...	{Q11471, Q208788, Q7017933, Q48282, Q833163, Q...	[0.17345399698340874, 0.04708097928436911, 0.0...
93	{Q44946, Q327968, Q862131, Q205, Q170475, Q370...	{Q2995644, Q218005, Q133250, Q68, Q188860, Q21...	{Q44946, Q205, Q37038, Q638, Q9418, Q33456, Q2...	{Q11471, Q208788, Q7017933, Q48282, Q833163, Q...	[0.17345399698340874, 0.2683615819209039, 0.39...
94	{Q44946, Q327968, Q205, Q2995644, Q9332, Q8450...	{Q2995644, Q218005, Q133250, Q68, Q188860, Q21...	{Q44946, Q205, Q7603810, Q1468740, Q33456, Q20...	{Q11471, Q208788, Q7017933, Q48282, Q833163, Q...	[0.17345399698340874, 0.22504708097928436, 0.4...
...	...	...	...	...	...
764	{Q207961, Q182832, Q42848, Q838119, Q12525525,...	{Q739462, Q207961, Q42848, Q2995644, Q9332, Q8...	{Q207961, Q1132755, Q638, Q9465, Q1434913, Q11...	{Q6717763, Q739462, Q9081, Q207961, Q156901, Q...	[0.12820512820512822, 0.1224105461393597, 0.26...
765	{Q327968, Q1479995, Q42848, Q312, Q5441226, Q1...	{Q739462, Q207961, Q42848, Q2995644, Q9332, Q8...	{Q22656, Q638, Q9418, Q312, Q18967146, Q228200...	{Q6717763, Q739462, Q9081, Q207961, Q156901, Q...	[0.12820512820512822, 0.31544256120527303, 0.2...
766	{Q11471, Q932615, Q922203, Q145874, Q12525525,...	{Q739462, Q207961, Q42848, Q2995644, Q9332, Q8...	{Q11471, Q932615, Q7748, Q204, Q4661718, Q8078...	{Q6717763, Q739462, Q9081, Q207961, Q156901, Q...	[0.12820512820512822, 0.07062146892655367, 0.1...
767	{Q327968, Q22687, Q2995644, Q131841, Q669094, ...	{Q739462, Q207961, Q42848, Q2995644, Q9332, Q8...	{Q669094, Q3286102, Q204, Q8078, Q370502, Q430...	{Q6717763, Q739462, Q9081, Q207961, Q156901, Q...	[0.12820512820512822, 0.04708097928436911, 0.0...
768	{Q42848, Q2995644, Q868299, Q507810, Q7561, Q1...	{Q739462, Q207961, Q42848, Q2995644, Q9332, Q8...	{Q1707206, Q11471, Q2607780, Q1351452, Q188830...	{Q6717763, Q739462, Q9081, Q207961, Q156901, Q...	[0.12820512820512822, 0.07344632768361582, 0.1...

670 rows × 5 columns

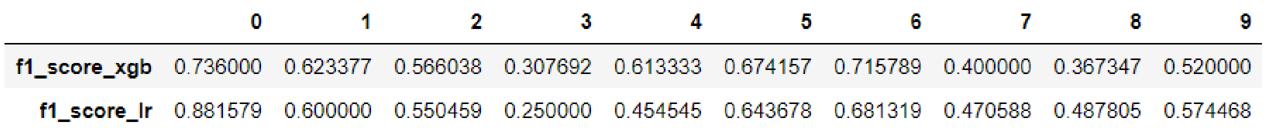
Figure 5.8: a presentation of obtained E_{ts} , E_v, I_{ts} *and* I_v, and the calculated feature vectors.

From the figure above, we can see that all the E_{ts} and I_{ts} related to the same query (chapter title) should be absolutely the same. In the meanwhile, E_v and I_v can be changed a lot according to the different retrieved videos related to the same query, which leads to quite different feature vectors.

5.3 Machine learning model training for cross-media retrieval

After getting the feature vectors as the inputs of machine learning models, the next step is to train classifiers which can discriminate whether a pair of cross-media resources are matched or not. Then among many different machine learning classifiers, we mainly trained logistic regression and Xgboost. And as introduced previously, since the scale of our dataset is really limited, we create 10 different datasets from the original dataset, which means that regard all the samples related to the queries from each book as the test sets, and all the other samples related to the queries from the other books as the training sets. And then train 10 different models to predict the specific test set of samples. For example, there are totally 10 books, for each book there are 10 different queries, and for each query there are 10 video results. Thus, after dropping several bad cases of samples (since some YouTube videos don't have subtitles), the scale of each train set is around 800, and the scale of each test set is around 90.

And among all the machine learning models that can be used as classifiers to do the classification, the reason why logistic regression and Xgboost have been chosen to do the experiment is that they are the most popular classifiers both in the field of machine learning competitions and industrial IT companies, and they are very simple to be utilized. Then we train 10 models of both logistic regression and Xgboost (cross validation has been done for finding out the best parameters) separately, and we can figure out that their performances are not really different.



	0	1	2	3	4	5	6	7	8	9
f1_score_xgb	0.736000	0.623377	0.566038	0.307692	0.613333	0.674157	0.715789	0.400000	0.367347	0.520000
f1_score_lr	0.881579	0.600000	0.550459	0.250000	0.454545	0.643678	0.681319	0.470588	0.487805	0.574468

Figure 5.9: comparison of the metric f1_score (a combination of the metric precision and the metric recall) between logistic regression and Xgboost models on 10 different constructed datasets.

And since Xgboost provides a utility called feature importance, which enable the visualizable analysis of the importance or effectiveness of each feature in the procedure of the classification, Xgboost models have been chosen to be utilized as the final machine learning models.

Then the feature importance maps will be presented as the followings:

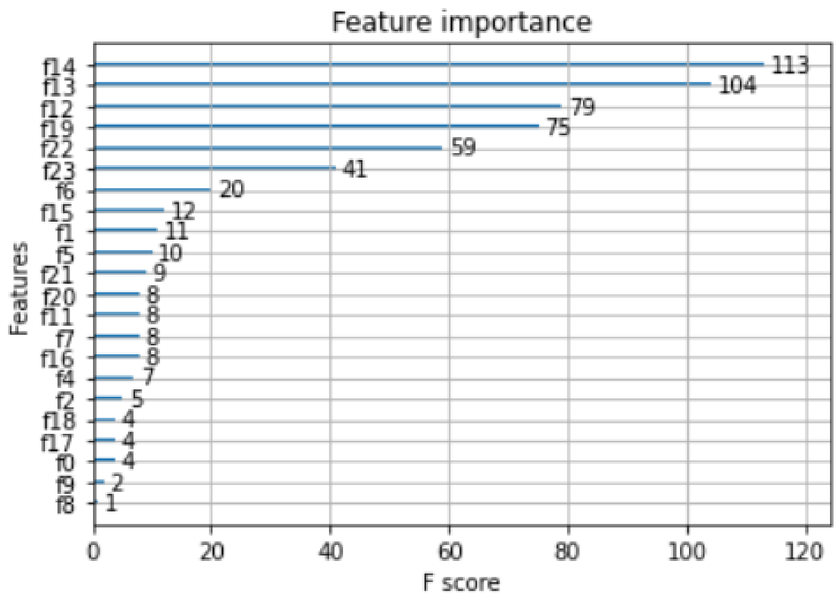


Figure 5.10: feature importance of Xgboost model on dataset 1.

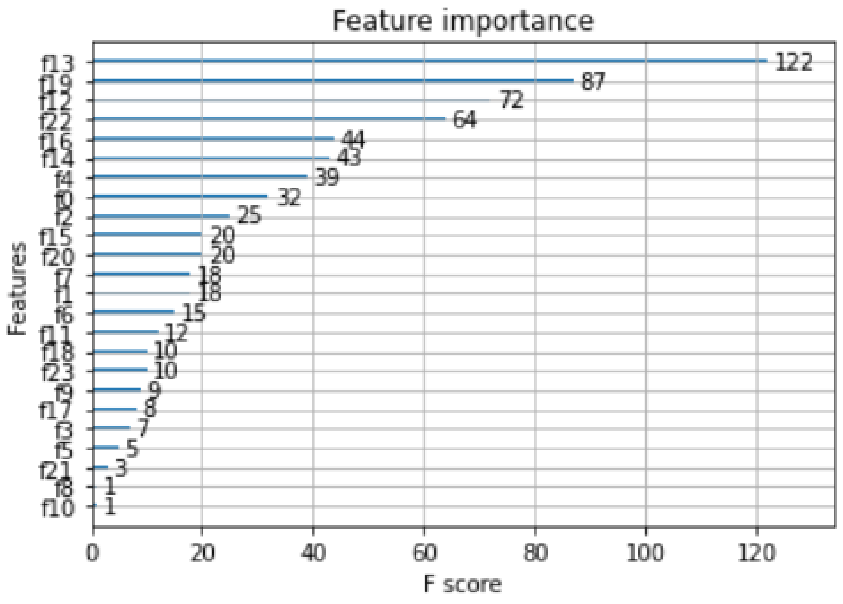


Figure 5.11: feature importance of Xgboost model on dataset 2.

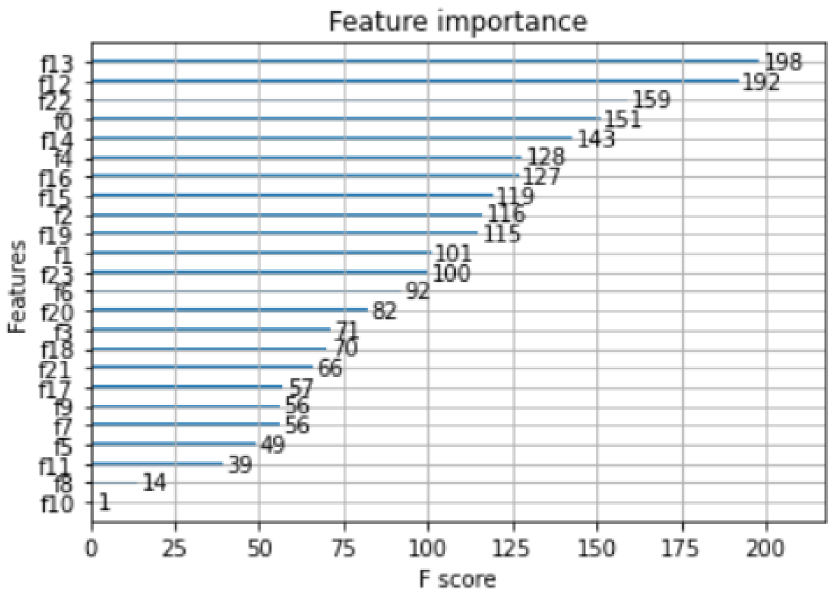


Figure 5.12: feature importance of Xgboost model on dataset 3.

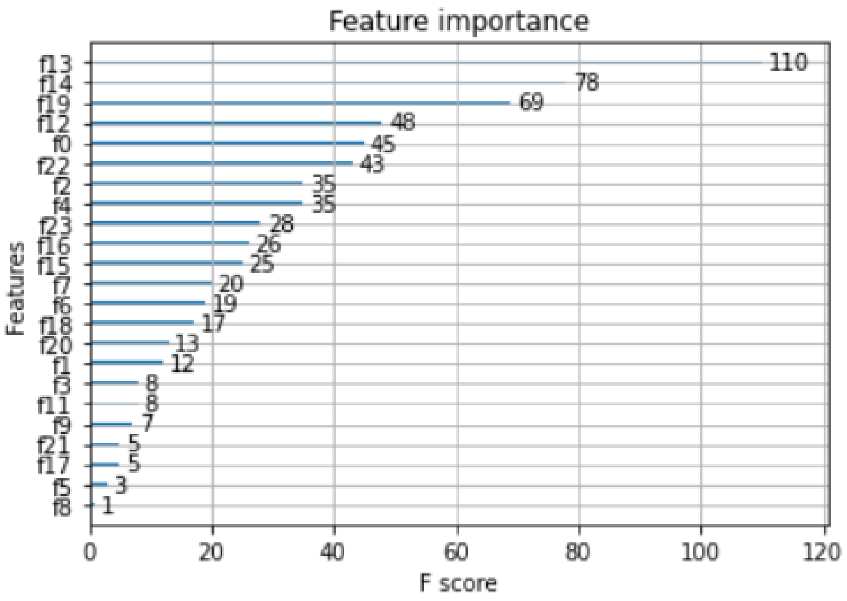


Figure 5.13: feature importance of Xgboost model on dataset 4.

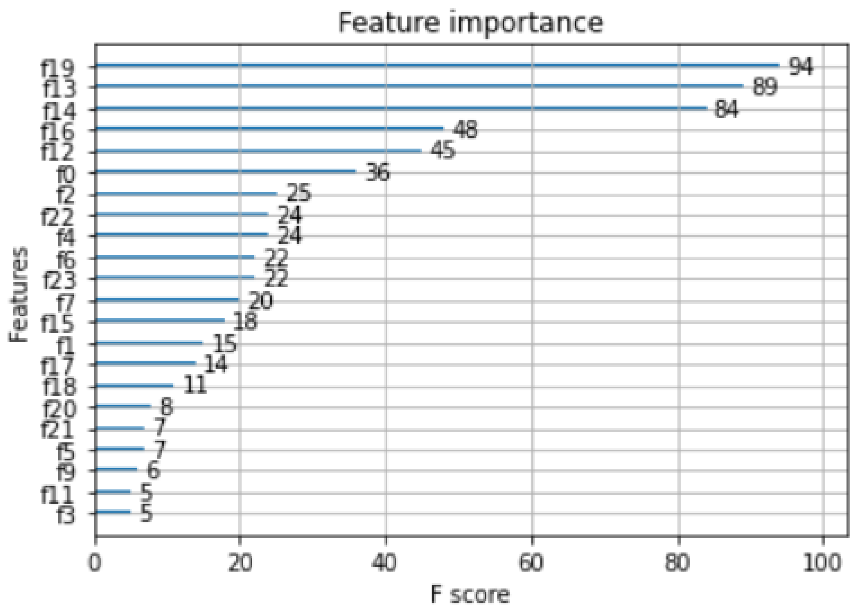


Figure 5.14: feature importance of Xgboost model on dataset 5.

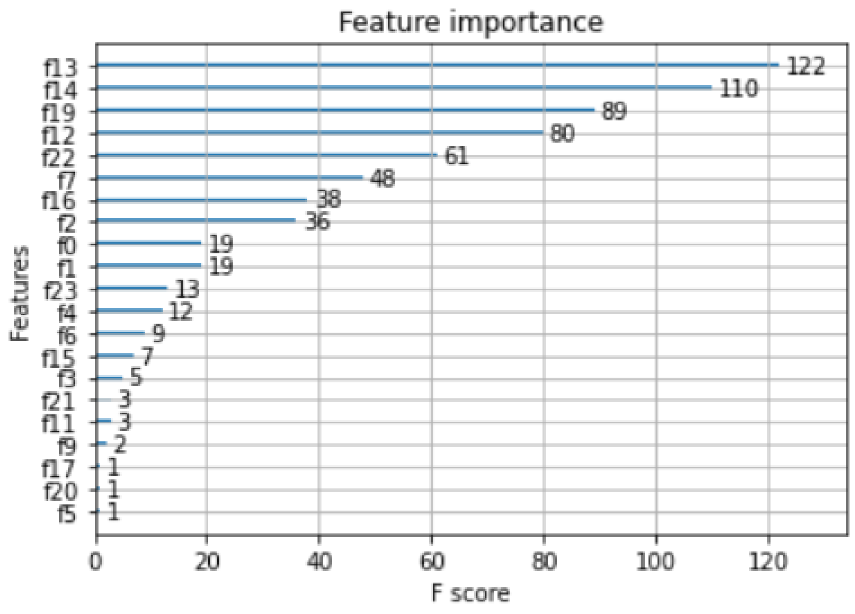


Figure 5.15: feature importance of Xgboost model on dataset 6.

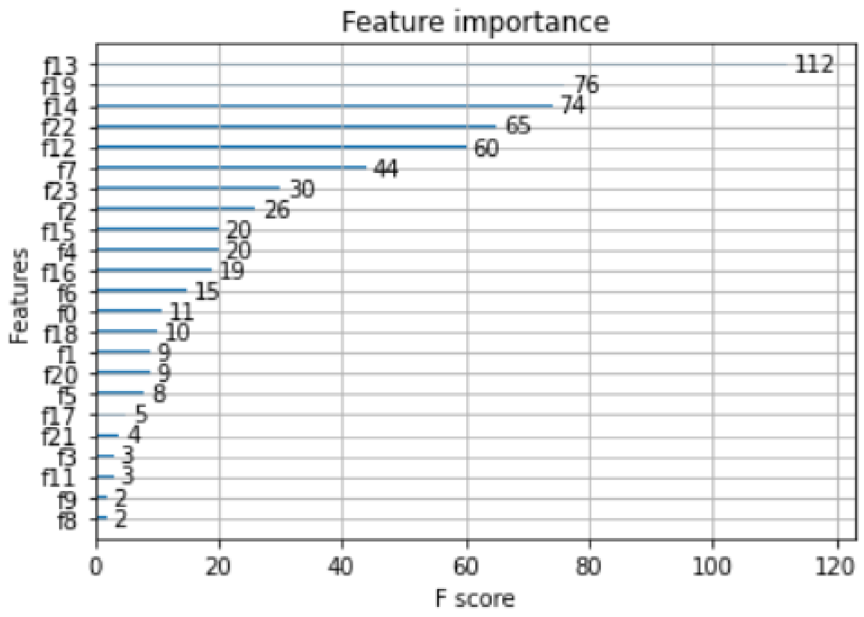


Figure 5.16: feature importance of Xgboost model on dataset 7.

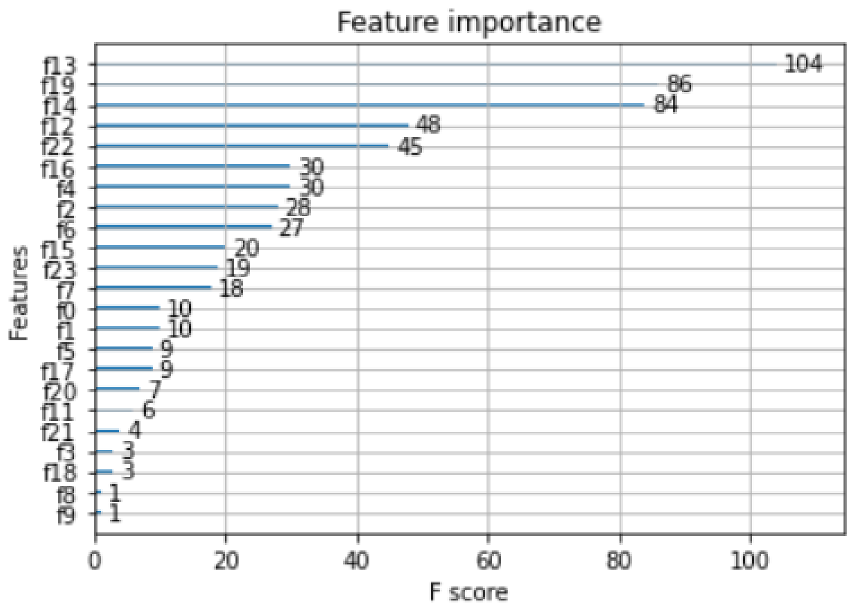


Figure 5.17: feature importance of Xgboost model on dataset 8.

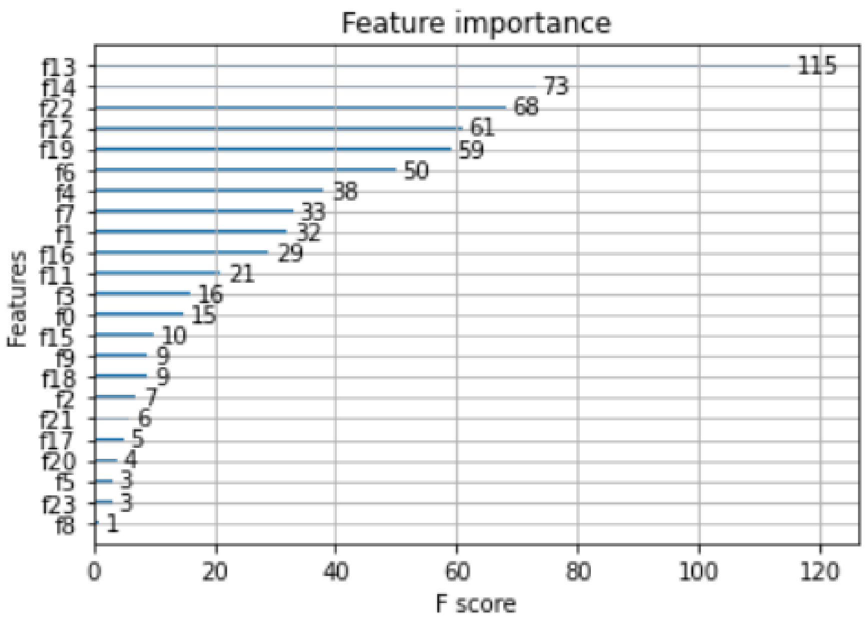


Figure 5.18: feature importance of Xgboost model on dataset 9.

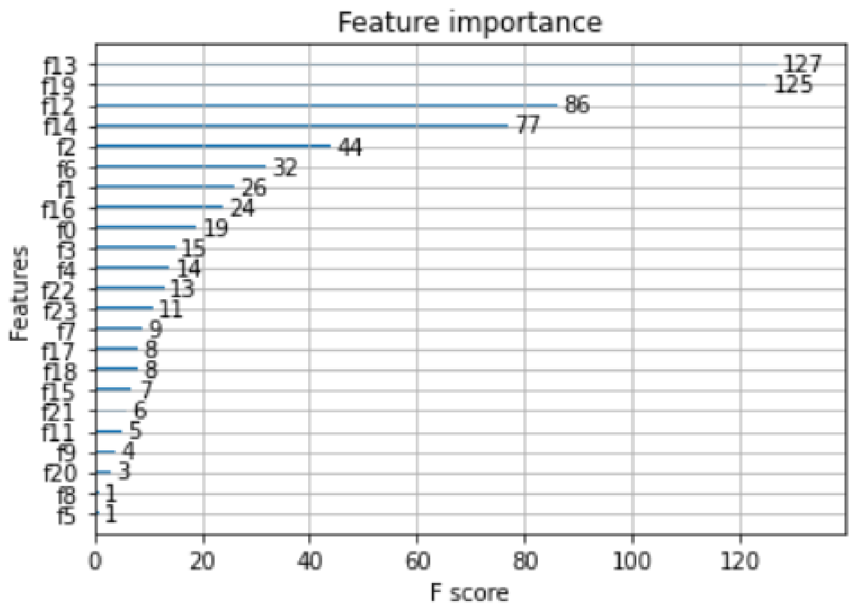
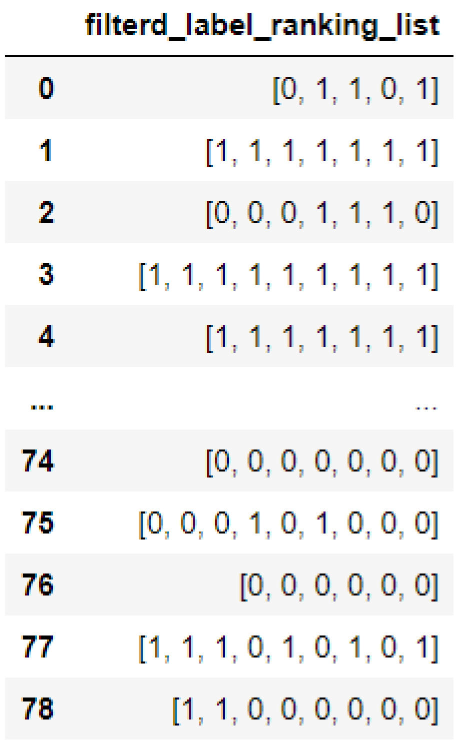


Figure 5.19: feature importance of Xgboost model on dataset 10.

From all the figures of feature importance above, we can easily figure out that there is a strong consistency in the ranking of feature importance with the 10 different Xgboost models trained on the 10 different constructed datasets. More in details, the features f1, f2, f6, f12, f13, f14, f19 always have been ranked as the most important features, and we can easily check out which features are referenced by these feature indices.

After the analysis of feature importance, during the procedure of predicting samples in different test sets, we also store the results of predictions into the data-frame of each test set, and 10 different data-frames with the prediction has been recombined together. Then we filtered out the rows with negative prediction since the trained model regarded them as not matched video results.

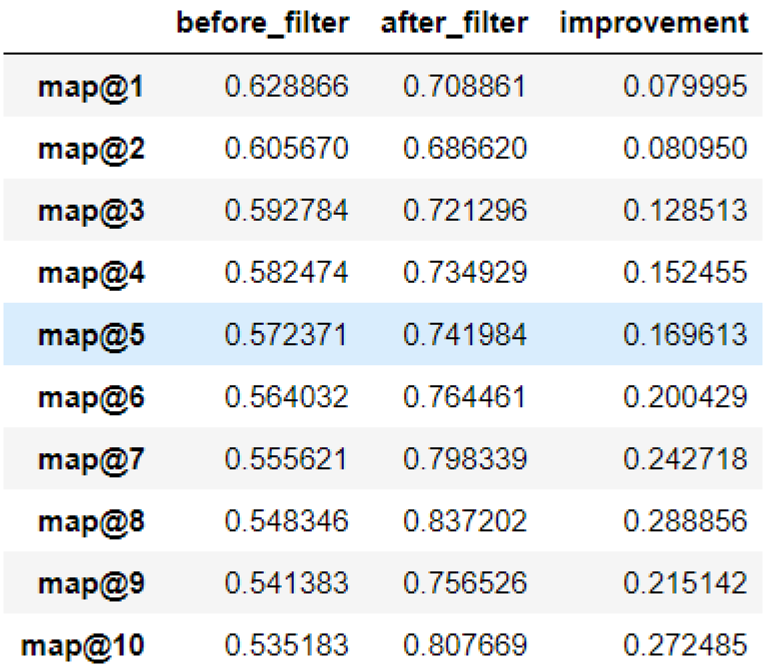


	filterd_label_ranking_list
0	[0, 1, 1, 0, 1]
1	[1, 1, 1, 1, 1, 1, 1]
2	[0, 0, 0, 1, 1, 1, 0]
3	[1, 1, 1, 1, 1, 1, 1, 1, 1]
4	[1, 1, 1, 1, 1, 1, 1]
...	...
74	[0, 0, 0, 0, 0, 0, 0]
75	[0, 0, 0, 1, 0, 1, 0, 0, 0]
76	[0, 0, 0, 0, 0, 0]
77	[1, 1, 1, 0, 1, 0, 1, 0, 1]
78	[1, 1, 0, 0, 0, 0, 0, 0]

Figure 5.20: result label ranking list of the queries after filter out by Xgboost classifiers.

Since we want to assess whether the quality of the ranking lists for each query have been improved or not, a metric introduced in chapter 3 called mean average precision has been calculated before and after filter by Xgboost classifiers. And then we compare the calculated values of mean average @ k (k=1, 2, ..., 10) for the assessment of our work globally.

Here is a figure of the comparison between calculated mean average precison @ k (k=1, 2, ..., 10) and the improvement after the filter by Xgboost classifiers:



	before_filter	after_filter	improvement
map@1	0.628866	0.708861	0.079995
map@2	0.605670	0.686620	0.080950
map@3	0.592784	0.721296	0.128513
map@4	0.582474	0.734929	0.152455
map@5	0.572371	0.741984	0.169613
map@6	0.564032	0.764461	0.200429
map@7	0.555621	0.798339	0.242718
map@8	0.548346	0.837202	0.288856
map@9	0.541383	0.756526	0.215142
map@10	0.535183	0.807669	0.272485

Figure 5.21: comparison between calculated mean average precison @ k (k=1, 2, ..., 10) and the improvement after the filter by Xgboost classifiers

By contrasting the two map @ k results, we can easily see that after filter out not matched video results, the rankings of the left video results for each query have been improved obviously. Especially in practice when the numbers of returned search results are much larger than 10, then both the quantity and the quality of the search can be guaranteed. But whatever, for a certain ranking list of the returned search results, the top 1, top 5, top 10 results are always very import as a measurement of the quality of search.

Until this point, after the construction of the data pipeline of cross-media resources introduced in chapter 4, and then cross-media retrieval by using Xgboost classifiers presented in this chapter, the whole procedure of the proposed NLP based cross-media retrieval has been implemented and explained. And finally, the improvement of all the mean average precision @ k (k=1, 2, ..., 10) can be a strong proof that the proposed method in this thesis is capable of being a prototype of an annotation system which annotates text (chapter titles, keywords and concepts) in PDF books with YouTube videos. Moreover, in the future, videos can be not only from a single video platform like YouTube, but also Coursera, Udacity and others.

Chapter 6

Conclusion and Contributions

From the most original purpose of providing a service which annotates the key concepts in books with YouTube videos, we did the work step by step, and finally get to a relatively satisfying point.

And there are mainly two different contributions of what we've done. The first one is from engineering perspective, the concrete engineering steps are:

1. Collection of books as resource of text query.
2. Randomly select 10 chapter titles from each book as queries.
3. Retrieve top 10 video results from YouTube automatically.
4. Implement a front-end representation which enable the direct representation to future users to see the relevant videos of text in books.

And the significant meaning of the engineering part is the automation of the process, which means it is feasible to design a similar but more end-to-end and robust system to provide the same kind of information service with high quality. And that's a requirement must be fulfilled if an engineering prototype project can be generalized as an educational or industrial service.

The second contribution is from information retrieval or application of machine learning algorithms perspective, the concrete steps are:

1. Extract sets of named entities from the text query in book and subtitles of returned result videos.
2. Construct subsets of named entity sets called instance sets.
3. Design and construct 24-dimensional feature vectors for each pair of text query and YouTube video.
4. Train Xgboost models to classify labeled datasets to determine the pair is matched or not.
5. Predict with test sets and filter out predicted not matched video results.
6. Calculate mean average precision of all the queries and compare it with the mean average precision of original ranking list from YouTube to check whether the machine learning filter works well.

And the significant meaning of the information retrieval part is the feasibility of continuously improvement of search quality, which means besides the service has an end-to-end and robust system, and the searched relevant video results can be more and more matched to the specific text queries from books. As a result, this kind of service can be probable more and more helpful for the students or any kinds of readers since whenever they have some doubts on some words or concepts in a pdf or online book, this kind of service will provide relevant videos directly when they click the text in that book.

And in the machine learning model part, although there was a strong disadvantage on lacking of labeled data, we could still get a lower bound above 0.7 of f1-score and **obvious increases at least 0.1 on all mean average precision @k (k=1, 2, ..., 10)**. And in our experiment case, there was sometimes a problem that the machine learning model will filter out most of the returned video results which lead to lack of video results for some specific query. But whenever we don't only consider the top 10 results of YouTube, for example, we make the pool of choices top 20 results of YouTube or even not only search in YouTube, also in some other video platforms such as Coursera, Udacity, then this problem will be solved properly.

As a conclusion, this work contributes both on the engineering aspect to provide a prototype of an annotation system of text in books with videos and on the cross-media retrieval algorithm aspect to provide a way to extract features and train machine learning models which filter out not matched results.

REFERENCES

[1] International Association of Universities, “Iau global survey on the impact of covid-19 on higher education around the world,” 2020. [Online].

[2] UNESCO IESALC, “Covid-19 and higher education: today and tomorrow. impact analysis, policy responses and recommendations,” 2020. [Online].

[3] O. B. Adedoyin and E. Soykan, “Covid-19 pandemic and online learning: the challenges and opportunities,” Interactive Learning Environments, pp. 1–13, 2020. [Online].

[4] R. Kay, “Exploring the use of video podcasts in education: A comprehensive review of the literature,” Computers in Human Behavior, vol. 28, pp. 820–831, 05 2012.

[5] E. Bravo, B. Garc´ıa, P. Simo, M. Enache, and V. Fernandez, “Video as a new teaching tool to increase student motivation,” 05 2011, pp. 638 – 642.

[6] Y. Peng, X. Huang, and Y. Zhao, “An overview of cross-media retrieval: Concepts, methodologies, benchmarks, and challenges,” IEEE Trans. Circuits Syst. Video Technol., vol. 28, no. 9, pp. 2372–2385, 2018. [Online].

[7] D. Rao and B. McMahan, Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. O’Reilly Media, 2019. [Online].

[8] T. Duffy and D. Jonassen, Constructivism and the Technology of Instruction: A Conversation. Taylor & Francis, 2013. [Online].

[9] P. Guo, J. Kim, and R. Rubin, “How video production affects student engagement: An empirical study of mooc videos,” 03 2014, pp. 41–50.

[10] P. Kaur, H. S. Pannu, and A. K. Malhi, “Comparative analysis on cross-modal information retrieval: A review,” Computer Science Review, vol. 39, p. 100336, 2021. [Online].

[11] Y. Peng, X. Huang, and J. Qi, “Cross-media shared representation by hierarchical learning with multiple deep networks,” in International Joint Conference on Artificial Intelligence (IJCAI), 2016, pp. 3846– 3853.

[12] F. Yan and K. Mikolajczyk, “Deep correlation for matching images and text,” in IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. IEEE Computer Society, 2015, pp. 3441–3450.

[13] S. Liu, S. Qian, Y. Guan, J. Zhan, and L. Ying, “Joint-modal distribution-based similarity hashing for large-scale unsupervised deep cross-modal retrieval,” in Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020, J. Huang, Y. Chang, X. Cheng, J. Kamps, V. Murdock, J. Wen, and Y. Liu, Eds. ACM, 2020, pp. 1379–1388.

[14] F. Feng, X. Wang, and R. Li, “Cross-modal retrieval with correspondence autoencoder,” in ACM International Conference on Multimedia (ACM MM), 2014, pp. 7–16.

[15] S. ur Rehman, M. Waqas, S. Tu, A. Koubaa, O. ur Rehman, J. Ahmad, M. Hanif, and Z. Han, “Deep learning techniques for future intelligent cross-media retrieval,” CoRR, vol. abs/2008.01191, 2020.

[16] H. Tong, J. He, M. Li, C. Zhang, and W.-Y. Ma, “Graph based multi-modality learning,” in Proceedings of the 13th Annual ACM International Conference on Multimedia, ser. MULTIMEDIA ’05. New York, NY, USA: Association for Computing Machinery, 2005, p. 862–871.

[17] F. Wu, X. Lu, J. Song, S. Yan, Z. M. Zhang, Y. Rui, and Y. Zhuang, “Learning of multimodal representations with random walks on the click graph,” IEEE Trans. Image Process., vol. 25, no. 2, pp. 630–642, 2016.

[18] X. Zhai, Y. Peng, and J. Xiao, “Heterogeneous metric learning with joint graph regularization for cross-media retrieval,” in Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA, M. desJardins and M. L. Littman, Eds. AAAI Press, 2013.

[19] G. Xu, X. Li, and Z. Zhang, “Semantic consistency cross-modal retrieval with semi-supervised graph regularization,” IEEE Access, vol. 8, pp. 14 278–14 288, 2020.

[20] X. Zhai, Y. Peng, and J. Xiao, “Effective heterogeneous similarity measure with nearest neighbors for cross-media retrieval,” in Advances in Multimedia Modeling - 18th International Conference, MMM 2012, Klagenfurt, Austria, January 4-6, 2012. Proceedings, ser. Lecture Notes in Computer Science, K. Schoeffmann, B. Merialdo, ´ A. G. Hauptmann, C. Ngo, Y. Andreopoulos, and C. Breiteneder, Eds., vol. 7131. Springer, 2012, pp. 312–322.

[21] D. Ma, X. Zhai, and Y. Peng, “Cross-media retrieval by clusterbased correlation analysis,” in IEEE International Conference on Image Processing, ICIP 2013, Melbourne, Australia, September 15-18, 2013. IEEE, 2013, pp. 3986–3990.

[22] F. Wu, X. Lu, Z. Zhang, S. Yan, Y. Rui, and Y. Zhuang, “Cross-media semantic representation via bi-directional learning to rank,” in ACM Multimedia Conference, MM ’13, Barcelona, Spain, October 21-25, 2013, A. Jaimes, N. Sebe, N. Boujemaa, D. Gatica-Perez, D. A. Shamma, M. Worring, and R. Zimmermann, Eds. ACM, 2013, pp. 877–886.

[23] F. Wu, X. Jiang, X. Li, S. Tang, W. Lu, Z. Zhang, and Y. Zhuang, “Cross-modal learning to rank via latent joint representation,” IEEE Trans. Image Process., vol. 24, no. 5, pp. 1497–1509, 2015.

[24] J. Tang, Z. Li, M. Wang, and R. Zhao, “Neighborhood discriminant hashing for large-scale image retrieval,” IEEE Trans. Image Process., vol. 24, no. 9, pp. 2827–2840, 2015.

[25] F. Zhong, Z. Chen, and G. Min, “Deep discrete cross-modal hashing for cross-media retrieval,” Pattern Recognition, vol. 83, pp. 64 – 77, 2018.

[26] F. Yan and K. Mikolajczyk, “Deep correlation for matching images and text,” in IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. IEEE Computer Society, 2015, pp. 3441–3450.

[27] H. Bhuiyan, J. Ara, R. Bardhan, and M. R. Islam, “Retrieving youtube video by sentiment analysis on user comment,” in 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), 2017, pp. 474–478.

[28] V. Gabeur, C. Sun, K. Alahari, and C. Schmid, “Multi-modal transformer for video retrieval,” in Computer Vision – ECCV 2020, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 214–229.

[29] M. Hadjieleftheriou and D. Srivastava, “Weighted set-based string similarity,” IEEE Data Eng. Bull., vol. 33, pp. 25–36, 2010.

[30] V. M K and K. K, “A survey on similarity measures in text mining,” Machine Learning and Applications: An International Journal, vol. 3, pp. 19–28, 03 2016.