SN Computer Science (2022) 3:33
https://doi.org/10.1007/s42979-021-00843-x

ORIGINAL RESEARCH

SmartEyes: Social Multimedia Analysis Platform for Open Data Providers

Thanh-Cong Le[1,2,3] · Quoc-Vuong Nguyen[1,3] · Minh-Triet Tran[1,2,3]

Received: 10 June 2021 / Accepted: 7 July 2021 / Published online: 1 November 2021

Abstract
It is essential to collect and analyze information from various data sources to develop smart cities. A huge amount of social media data are generated daily from various social activities in different formats, such as text, images, audio, or video clips. Thus, we aim to develop SmartEyes, a flexible platform for collecting, integrating, and analyzing social multimedia for open data providers for smart cities and environments. Our platform SmartEyes consists of useful components, which can be easily extended integrated to quickly develop different social media analysis services to listen and analyze data from different social media sources with the diversification of data types. We emphasize the ability to flexibly integrate artificial intelligence applications into the system to analyze social events effectively and serve smart cities in creating open data providers. We also introduce four case study applications based on our platform, including a face recognition system for celebrity recognition in news videos, an object detection system for brand logo recognition, a video highlighting system for summarizing football matches, and a text analysis system serving for keyword occurrences and emotional text analysis for admissions of universities. In these applications, we have collected and analyzed more than 5000 videos from various Youtube channels, and thousands of posts from admission pages of universities on Facebook. Each application demonstrates a unique meaning to each specific situation for open data providers in smart cities.

Keywords Social listening · Open data provider · Software architecture · Object detection · Face recognition · Video highlights

Introduction

Developing smart cities has become an inevitable trend worldwide to apply intelligent technologies and applications to assist people in various daily activities. Smart cities enable technological and social innovation for fostering productivity, sustainability, and livability [14].

One of the key components of developing a smart city is creating open data providers to collect and analyze data to help us understand and infer information from society and the environment. This leads to the urgent need to listen and analyze social multimedia data for open data providers in smart cities.

The development of media, as well as devices to record data in daily activities, has created a rich and diverse data source in both genre and quantity [18]. Multimedia data can be collected or generated from different devices, e.g. smartphones, security cameras, traffic cameras, personal cameras, or other recording devices. Moreover, such data are not only in text but also in other formats [4], e.g., audio, image, video, providing us meaningful insights about personal and social impacts.

Listening and analyzing systems on social multimedia are of great interest in the world as well as in Vietnam. Typically, such systems rely on periodic and regular crawling

This article is part of the topical collection "Future Data and Security Engineering 2020" guest edited by Tran Khanh Dang.

✉ Minh-Triet Tran
tmtriet@fit.hcmus.edu.vn

Thanh-Cong Le
ltcong@selab.hcmus.edu.vn

Quoc-Vuong Nguyen
nqvuong@selab.hcmus.edu.vn

1 University of Science, VNU-HCM, Ho Chi Minh City, Vietnam

2 John von Neumann Institute, VNU-HCM, Ho Chi Minh City, Vietnam

3 Vietnam National University, Ho Chi Minh City, Vietnam

to pull information from various sources such as websites, communities, forums, social networks, YouTube channels, and comments where people mention a certain brand, organization, or person, especially a celebrity. Analyzing these factors helps people know what information is talked about a lot, trends in society, uncover problems or potential problems related to a brand, individual, or organization [4]. Therefore, many social multimedia listening and analysis systems have been developed to focus on listening, collecting, analyzing, and making reports so that individuals and organizations can take appropriate actions.

The features of social multimedia analysis systems are increasingly added and enhanced by researchers and enterprises to help us analyze social media data better, instead of initially including only functions such as keywords statistics [4]. A solution based solely on text analysis may not provide a holistic view and not taking advantage of the diversity of data types and sources. Images, audio, and video clips are considered as an indispensable source for social media analysis [10].

From the above analysis, it is certainly that listening and analyzing social multimedia data are of great interest and has the strong potential to be widely applied in life more and more. This motivates us to develop SmartEyes, a flexible smart social multimedia analysis platform that can easily create open data providers for a smart society. We introduced the first implementation of our social analysis platform in [15]. In this paper, we present in detail the architecture of the enhanced version of our system.

We focus on the three essential principles in developing our smart social multimedia analysis platform as follows:

1. Wide variation of data formats: social media data is not only in text form but also in the audio, image, and video formats. Hence, our smart social media analysis platform should be flexible to handle many different data formats.
2. Wide variation of analysis functions: the analysis features are not limited to those related to statistics, comparison, etc. As analysis features can be of great diversity, a smart social media analysis platform should easily integrate new analysis features, e.g., face detection, and recognition, special event detection, and object counting.
3. Service provider for different data sources: the smart social multimedia analysis platform is not necessarily associated with the data collection process because, in addition to the data that we can collect, much data have been collected with specific characteristics. Individuals or organizations also need services/APIs from the platform for their own solutions or systems.

Our SmartEyes system is not a fixed-feature application. Instead, our flexible platform provides a set of architectural mechanisms and components that can be used or integrated to create different concrete applications to analyze a particular media format with a set of desired analysis functions from a specific data source. With a specific requirement in a particular application, our toolkit and solutions can be used to create an application to serve that request quickly. However, because of the diverse and rich needs and new requirements that may arise later, the architecture must be flexible to pair, integrate to serve different needs, and, at the same time, expand in the future. Therefore, it is essential to provide a clear solution to guide users about the tool, and this solution can integrate and add other features according to actual requirements and specifications.

We developed four use cases to illustrate the applicability of the SmartEyes solution to different scenarios. In the first application, we analyze news videos from the CNN channel on Youtube for more than 6 months to determine when key persons, i.e. celebrities, appear based on face recognition, thereby visualizing the results of the news on the map. The second application focuses on detecting logos or brand names in TV shows for advertisement analysis. In the third application, we exploit audio information from video clips of football matches to detect and highlight special events. Finally, the fourth application demonstrates text analysis to monitor news articles.

The structure of this paper is as follows. In the next section, we briefly review existing methods and approaches for social listening, social media data analysis systems, and AI applications, which are used to provide intelligent suggestions or improvements. The third section presents in detail our proposal for overview architecture with the ability to flexibly integrate artificial intelligence analysis applications into the system based on using multiple data source management. We also developed four systems for case studies with their practical meanings, which are described in the fourth section. Finally, the conclusion is in the last section.

Related Work

The establishment and application of social media data analysis systems are issues organizations and businesses have focused on developing in recent years.

YouNet Group is a leader in Marketing and Technology, developed on the Social Intelligence platform. In 2013, the subsidiary YouNet Media was established and became one of the market leaders in social media information analysis thanks to technology applications and multichannel data sources, namely SocialHeat. In 2015, by owning Buzzmetrics, YouNet Group affirmed its leading position in social listening and data analysis market with over 70% market share. Besides, Viettel has also researched and built a Reputa platform to support the monitoring and analyzing social media information. Both SocialHeat's YouNet Group and Viettel's Reputa are quality products that bring positive signals that open up new market analysis trends for businesses—the trend of analyzing social media information.

In general, SocialHeat is a product that supports YouNet Media's monitoring and analysis of social media information—a member of the YouNet Group ecosystem—a pioneer in the application of social intelligence and technology to operations and regulations on marketing process that brings comprehensive digital steps and boosts business efficiency for customers in Vietnam and international markets. This is an integrated platform with a Sentiment Rating Engine [8, 21] with an accuracy of 80–85%. With SocialHeat, businesses can monitor brand status in the media, detect and warn brand crises, identify those who harm the brand, identify people used for branding, identifying brand attributes on social networks, what topics are of interest, and brand-related trends, and analyzing trends over time.

Besides, Reputa is one of the pioneering platforms of Viettel to support social media information analysis. The Reputa system supports business organizations to collect and analyze information on social networks to help manage brands, capture needs, and take care of customers on the Internet and social networks. Reputa offers the main solutions: monitoring brand health daily through automatic nuances analysis, preventing communication crisis risk, understanding the market and competitors through tools support classification, and statistical market trends. Reputa also supports application of artificial intelligence technologies such as natural language processing (NLP) [4], automatic speech recognition (ASR) [9], and optical character recognition (OCR) [23].

One of the social listening system features is that it collects and can analyze that information to provide intelligent suggestions or improvements. With object detection, many algorithms are introduced, but the most common ones are Faster R-CNN [6], histogram of oriented gradients (HOG) [20], single shot detector (SSD) [27], YOLO [1], etc. With face recognition recently featured with FaceNet [26], Face detector using Dlib or OpenCV [5]. The algorithms have opened up many approaches that yield auspicious results.

Proposed Architecture for SmartEyes Platform

One of the important factors to develop smart cities is to have data providers in open form. Therefore, one of the essential needs is to create open data providers that can help us understand what social media information is available in the community. To do these quickly, it is essential to use utilities and components that can be easily integrated and extended, which motivates us to propose a platform with the appropriate components and utilities to help quickly create such systems.

In [15], we briefly introduced the first implementation of SmartEyes. This section presents details about the architecture of our enhanced system for social multimedia analysis.

Overview of System

This section presents our proposed method to provide a flexible platform, components, and utilities to crawl and analyze data from social networks with the diversification of data types such as text and image/video from multiple data sources such as Facebook, YouTube, and E-news. There are four main components in our method: gateway, crawler, statistics, and analysis (Fig. 1).

Gateway: We aim to build a support module that provides API services to communicate with clients with JWT authentication [2] and load balancing [7] ability to adjust traffic to the system. We present it in details in "Gateway".

Crawler: This module is provided to collect data from many social data sources such as Facebook, YouTube, or online newspapers like VnExpress. The details of the process of this module are described in "Crawler".

Statistics: The analysis system always needs data for statistics and visualization. After being collected in the crawler module, data are calculated and stored in a database like MongoDB. In addition, data need to be cached, and we use Redis for caching. This module does that task, and the process content is covered in "Statistics".

Analysis: The process of this module is illustrated in "Analysis", which is focused on how we can integrate different components from Google Colab [3] to create a smart API. Especially, it is described how we can break the gap between developing a regular distributed system with APIs and intelligent functions. In this one, we propose a simple, efficient architecture that can be used to convert a regular system, which can be deployed on Google Colab [3]. Besides, we also suggest a mechanism to communicate between different modules and services.

The approach of our proposed method emphasizes the ability to integrate artificial intelligence analysis applications into the system flexibly. Moreover, using artificial intelligence applications as a replaceable module according to user needs will bring extremely high scalability.

Gateway

Different applications can communicate and exchange data with each other via APIs. Typically, APIs are deployed on a server and handle user requests. However, when traffic increases, a server may not be able to handle all requests. Therefore, having more than one shared server handles user requests is an effective solution. At that time, it is

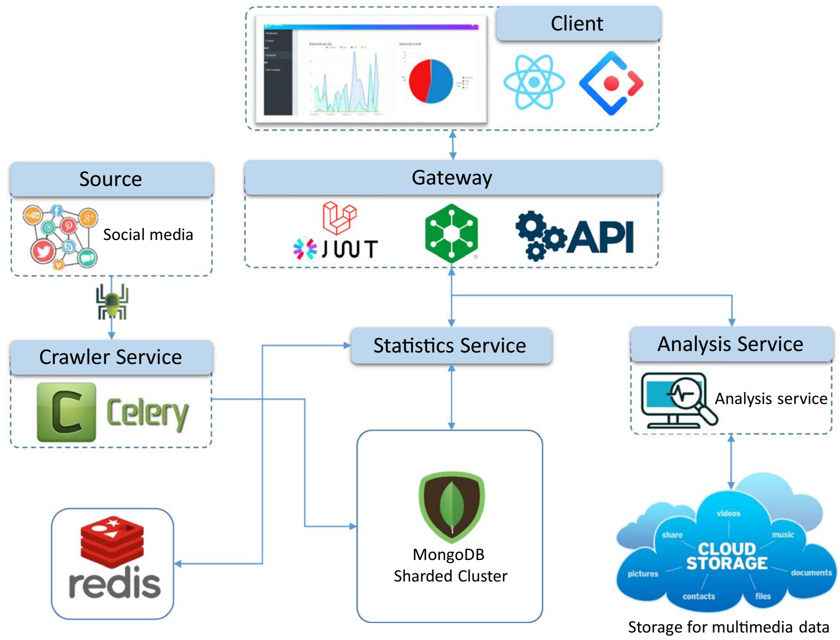


Fig. 1 Overview architecture of SmartEyes platform

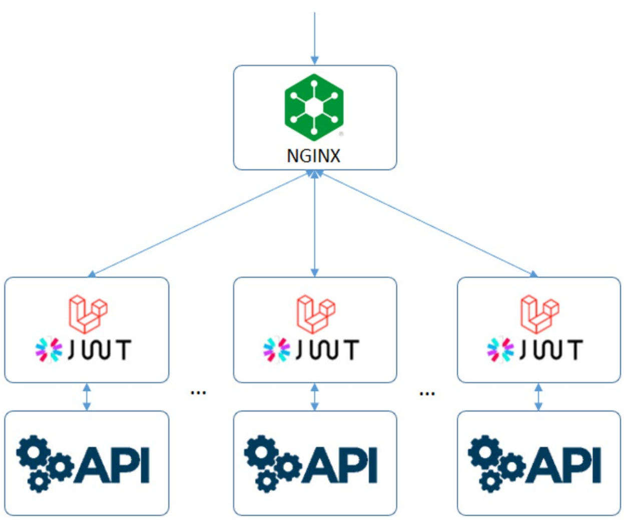


Fig. 2 Architecture of gateway

necessary to add a server that distributes this traffic to different servers logically in Fig. 2.

NGINX is a solution that supports load balancing between servers easily and efficiently [7]. NGINX receives all user access requests, then distribute those requests to a specific server.

In addition to load balancing, for users to be able to access the API on each server, access requests must go through the JWT [2] user authentication layer. User authentication is required to enable application users to use features within the application, namely to communicate with the API.

Crawler

Collecting data from social media such as news sites, and social networks helps to diversify sources of information, providing additional data to support analysis. However, data from social media is immense and varied. Therefore, a solution is needed to help gather data from many sources quickly and effectively. Moreover, the solution needs to be highly flexible to expand data collection from many other sources in the future.

Celery is a highly scalable, parallel task queue management system. Figure 3 shows the four steps of the Celery data collection process.

1. The task of collecting data from Facebook, YouTube, and VNExpress are declared and sent to the Celery client.
2. The Celery client distributes tasks into the RabbitMQ queues.
3. Depending on the Celery worker's status, the RabbitMQ queues' tasks are transferred to a certain Celery worker for processing.
4. The collected data are stored in MongoDB. Multimedia data such as images and videos are stored as strings, linking to their direct storage source.

According to the MongoDB Sharded Cluster model, the MongoDB database is implemented to help disperse a large amount of data while ensuring high throughput and low latency for efficient analysis. Data collection tasks are executed daily to ensure the latest information and data are updated to meet users' needs.

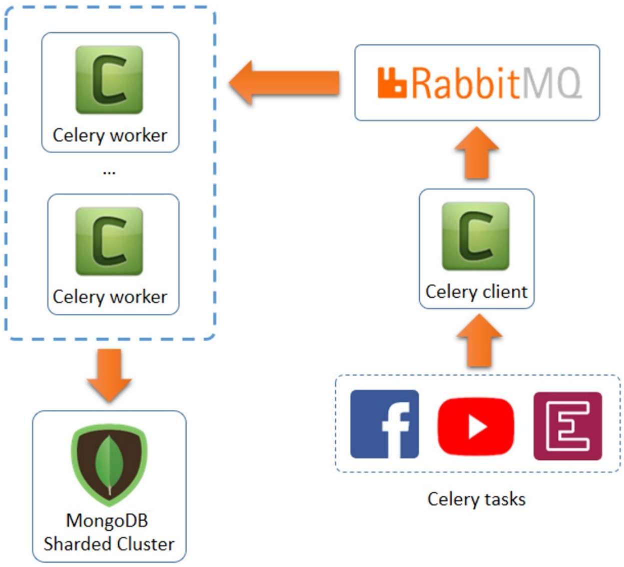


Fig. 3 Architecture of crawler

Statistics

This module is responsible for using collected data to calculate the metrics needed for visualization and also exposes the APIs for other services to use: information APIs supporting for details of sources and statistics APIs supporting for analysis metrics of sources.

The statistics module needs fast access efficiency and consistent storage capabilities, so combining a database for storage and another database for caching is needed.

MongoDB not only stores, updates, and retrieves information but also supports calculating data thanks to the Aggregation Framework. Through API calls into MongoDB, it is easy to retrieve the data for comparison and statistics. However, when the number of requests to access information and data increases, the use of the cache to store the results of frequent queries in the database is a necessary solution. Thus, to avoid querying too much into the database as well as quickly responding to query results.

We use Redis as a cache server to store the results of queries into MongoDB. The strategy of updating the cache when having a query like this:

1. Find data in the cache. If the data exist (cache hit), go to step 4. Otherwise, if the data do not exist (cache miss), continue to step 2.
2. Retrieve data from the database.
3. Add newly retrieved data to the cache.
4. Return retrieved data.

Hence, with the above cache update strategy, it is possible to avoid caching unnecessary data, but the data in the cache may be inaccurate when the database is updated. We solve this problem by using TTL (Time to Live) for the data cached.

Analysis

The overall architecture of a data analysis service is described in Fig. 4, which shows a general model for data analysis that includes both textual and multimedia data. Specifically, most API calls go through three stages: preprocessing, main processing, and postprocessing. The preprocessing stage has the main task of processing input inputs, saving necessary data into Storage. The main processing stage depends on the needs and purpose of the processing; for example, object detection uses the CenterNet [29] library, face recognition uses the Face recognition of Dlib [22] library and then load the model from Storage to process and return results. Finally, the postprocessing phase saves the results to Storage and finishes the processing. Data of the same format are converted to a common format and centrally stored in a Storage that can be easily saved and shared. We use MongoDB to store textual data and Google Drive to store multimedia data such as images and videos because we found the fit.

We can have many different features for a social media data analysis system. However, through the analysis of common needs, we first recommend integrating system analysis with the following common component for the artificial intelligent process. We focus on two main data types: text and video/image. Therefore, we currently integrate the following API into our system.

Text Analysis

First, for text processing, we incorporate systems capable of doing parts involving sentimental analysis. For English, we use the model that has been trained with the English dataset to integrate into the Sentimental analysis model in Fig. 5. For Vietnamese, we first use the Vietnamese sentiment analysis model, and we also study to use PhoBERT [19] in the future. This module is integrated and turned out to be a Sentiment analysis model of main processing in Fig. 5 to provide features to support positive or negative nuances analysis from user comments.

Besides, we also support counting keywords from textual data. Counting keyword occurrences is a basic feature to make statistics about the frequency of keywords we are interested in. The input text data are normalized by returning all characters to lowercase and deleting the bar marks. Then, use the dictionary provided to count the number of keyword occurrences.

Fig. 4 Overall architecture of an analysis service

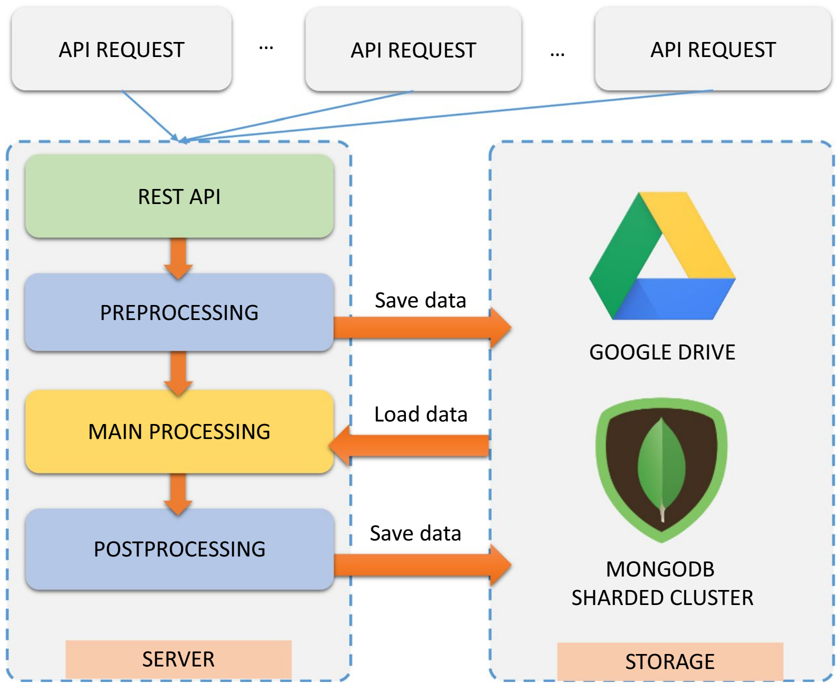
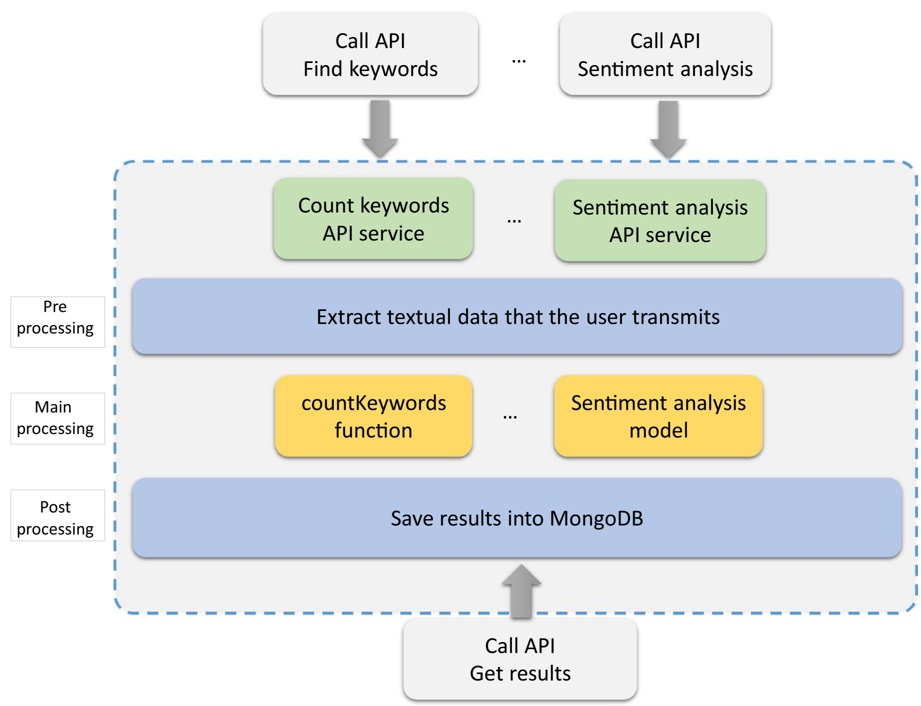


Fig. 5 Process of text analysis



Video/Image Analysis

For video/image data types, we focus on two common types of analysis: face recognition and object detection. In addition, we also consider features related to highlight videos [17]. To do this, we use the CenterNet algorithm for object detection, the Dlib library for face recognition, and the video volume solution for a video highlighting in Fig. 6. The procedure for each type of analysis is presented in the following part.

Object Detection

We use CenterNet for object detection because CenterNet is one of the leading new, fast, and accurate approaches to object detection in early 2020.

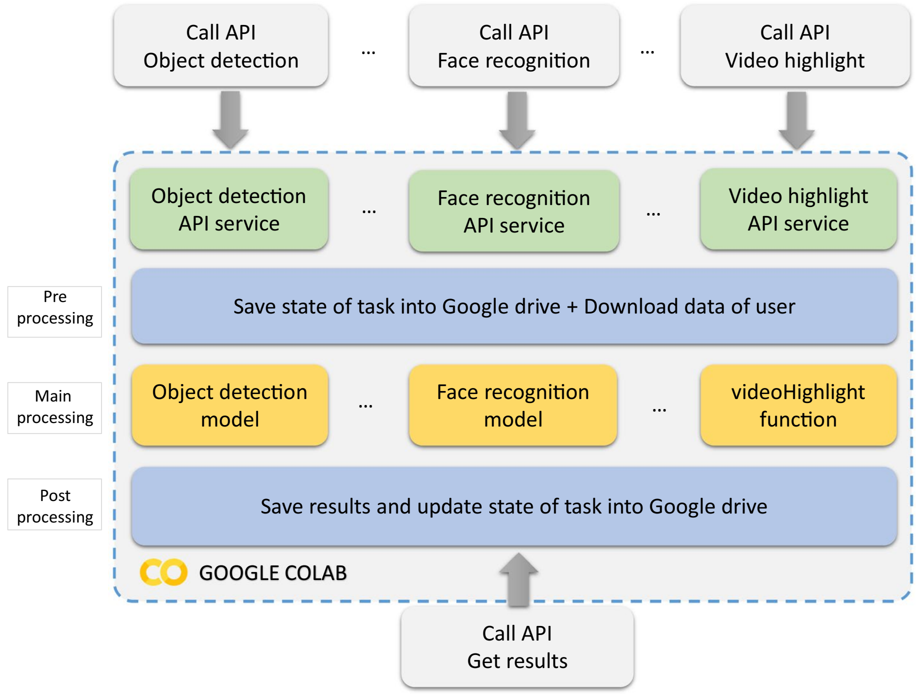


Fig. 6 Process of video/image analysis

CenterNet is an object detection network with a simple design but achieves balance both good speed and precision just released in 2019. CenterNet—Objects as Points [29] achieved the top results in the Realtime Object Detection problem on the COCO [16] dataset with quite high accuracy in early 2020. Specifically, 28.1% AP (average precision) at 142 FPS, 37.4% AP at 52 FPS, and 45.1% AP at 1.4 FPS using different backbone according to statistics from PapersWithCode [11].

The problem with most successful object detection networks today is that they have to cycle through all possible object locations and perform a classification for each of them. This leads to wasted computation resources, inefficiencies, and the need to do non-maximum suppression.

CenterNet's new approach brings the object detection problem to the keypoint estimation problem, thereby also deducing the size and calculating the bounding box for the object detection problem. The network architecture can also be easily modified to output 3D position, direction, and posture for other problems.

Using the CenterNet library, we can either use the pre-trained model or create a new personal model. To create a new model, the first thing is to prepare the labeled training set in the COCO [16] format. Then, we can store the label set data directly into the CenterNet folder on Cloud storage or compress it and upload it to a Google Drive, then pass the link to the API train new model shown in Fig. 7.

Figure 7 demonstrates the general process for adding new models that can be applied to both Object detection and Face

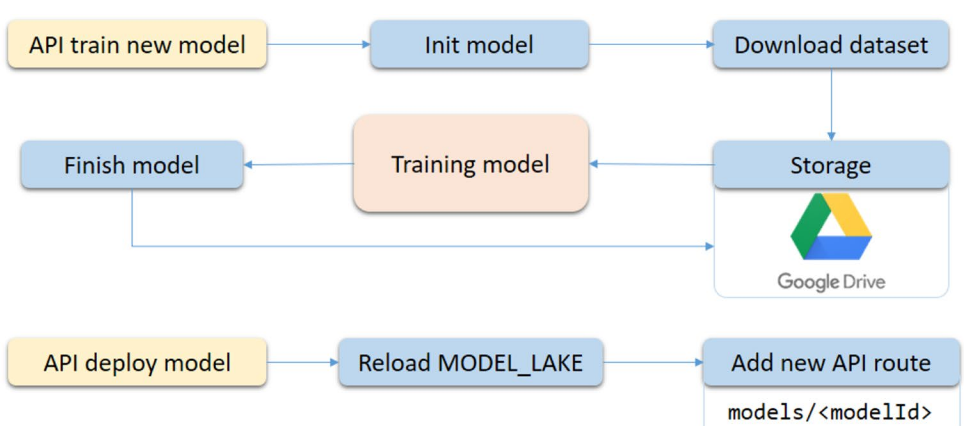


Fig. 7 Procedure of training new model

recognition with a similar way of just changing the Training model step with their respective tools. The process is mainly divided into two phases corresponding to the API to train new model and the API to deploy model. Still, the model has not been used only when deployed and creates a new URL for other services access. Then, the new exterior is actually applied. Information about the link dataset and model name are saved at the Init model to mark the processed model. Next, the Download dataset step relies on the link dataset to download and save to the CenterNet folder on Cloud storage like Google Drive. The Training model uses that data to train the new model. The training model uses the function of CenterNet library. So if we want to use another library or use Face recognition instead, we need to change this part. After the training model is complete, the Finish model saves it and its attached information to Cloud storage like Google Drive. However, after training the model, we cannot use it yet; we need to have to deploy step, load the model from Cloud storage, and add the URL for the API of the model that can be used with the path in Fig. 7.

As object detection and face recognition are both image-focused, images uploaded from Cloud storage can be analyzed directly. However, we need preprocessing for video clips to convert a video to images by cutting out frame by frame. We use FFmpeg [28] tool to do this transformation. After the video is preprocessed, object detection and face recognition input has been converted to a single image format in Fig. 8. The data are then processed through object detection models preloaded into the model lake on the server and stored in Cloud storage.

Face Recognition

We use the face_recognition [12] library, which was built using dlib's state-of-the-art face recognition built with deep learning by Adam Geitgey. The model has an accuracy of 99.38% on the Labeled Faces in the Wild benchmark [25]. In this paper, we do not focus on algorithm analysis but artificial intelligence functions usage and integration into the system. Thus we explore how to use the face recognition library. The face_recognition library with over 35k star ratings on Github for its accuracy and ease of use.

The face recognition training new model process is similar to object detection in Fig. 7. This library-based face training is divided into the following steps. First, prepare training data with a directory of subfolders that are the characters' names to be recognized for, and within those folders are face images. Note, the images used for training must be images with only one face. Then, when we install, we go through the folders in turn and use the face_recognition library to upload each image, marking the position of the face and the corresponding label. Then, we have an array of facial features in the form of vectors and labels, respectively. These data are further processed by scikit-learn's support vector machine [13] library, SVC, and saved as a model. Finally, the model is stored on Cloud storage and loaded up as needed.

Face recognition's video and image processing is similar to object detection in Fig. 8. However, the main processing uses the face recognition model preloaded from Cloud storage to the lake model.

Video Highlight

The approach of summarizing this video is not based on the training of models, visual event tracking, but the sound of the video. Most videos, not all, have a climax in the big soundtrack that is suddenly compared to the rest. For example, in football, every time a player scores, the commentator and the audience cheer loudly. Thus, audio tracking from a video can be handled using the moviepy.editor library in Python [30].

For example, the steps for highlighting videos are as follows. First, calculate the average volume per second of the audio portion from the video in Fig. 9. However, to see the differences in sound, the volume should be calculated at 10-s intervals. Next, to take the standout scenarios, we can take the occurrences of the top 10%. After that, all you need to do is cluster the occurrences no more than 1 min apart into the same group. As a result, we have event groups that have the volume in the top 10% of the video, and the rest is cutting the video according to those time groups.

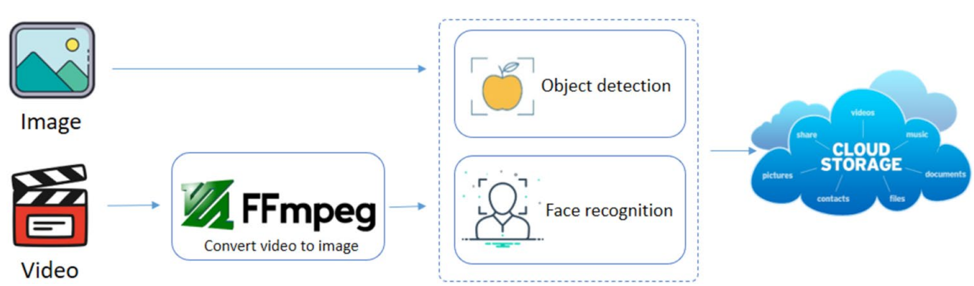


Fig. 8 Object detection and face recognition with video and image

Fig. 9 Volume per second of a football video

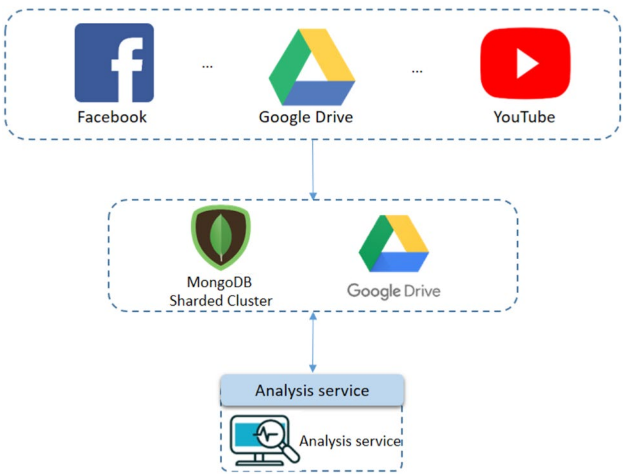


Fig. 10 Source provider

New API Service Integration

At present, we follow these APIs, but in the future, we can easily integrate more components by wrapping based on the architecture. Therefore, we briefly introduce the basic process to wrap an API into the system in this part. First, we deploy a template analysis service API on Google Colab [3] that includes preprocessing, main processing, and post-processing. Then, depending on the analysis needs, the developer can write custom wrapper functions to replace the contents of the above steps. Finally, the API is published and used by other services and modules.

Source Provider

Many data sources can be attributed to only two types of storage: database for text and cloud storage for video/image. The source provider we use is MongoDB and Google Drive in Fig. 10.

To manage information sources, we need to convert data from various formats to a single common one. Specifically, data collected automatically from social networks and user-uploaded data are stored on shared storage. This gives a prominent advantage to user-private data such as surveillance camera data, traffic cameras, personal photos, and videos.

When users have a dataset that does not allow direct access, and we can only use the exact data they provide, we can provide a simple solution for this scenario by uploading data on common cloud storage. At that time, we chose Google Drive to turn into cloud storage for storage purposes. However, other technologies, such as Amazon Drive, Dropbox, and OneDrive, can also be used.

For textual data and accompanying statistics, a database is suitable for storage. We use MongoDB and design a schema according to data source and username appropriately.

The management of data sources of various origins is addressed according to the storage directory hierarchy for multimedia data. Data folder on Cloud storage contains folders by data format names such as Image, Video. The Video folder contains subfolders according to the Username. Each Username folder contains subfolders that are the names of data source sources such as YouTube and Drive. These folders again contain Source ID folders. Inside the Source ID are the user data files and associated analysis files.

Communication and Integration Solution of API Services and Modules

The solution for communicating between API services and modules through sharing shared data in cloud storage. The specific procedure is shown in Fig. 11.

When the request is sent to the API service, the task is initiated and stores task status in the cloud storage as a text file. The request has general content, including video or image link, name task, username, data type, and data source. The server uses the link, data type, and data source to download data to the corresponding cloud storage. After that, the data are processed through the Analysis service and record the resulting path when End task. During the process from the Init task to the End task, users can invoke another API to get the status of a task to know whether it finishes or not. When the task is completed, the resulting data stored as a text file on cloud storage can be accessed through the resulting API.

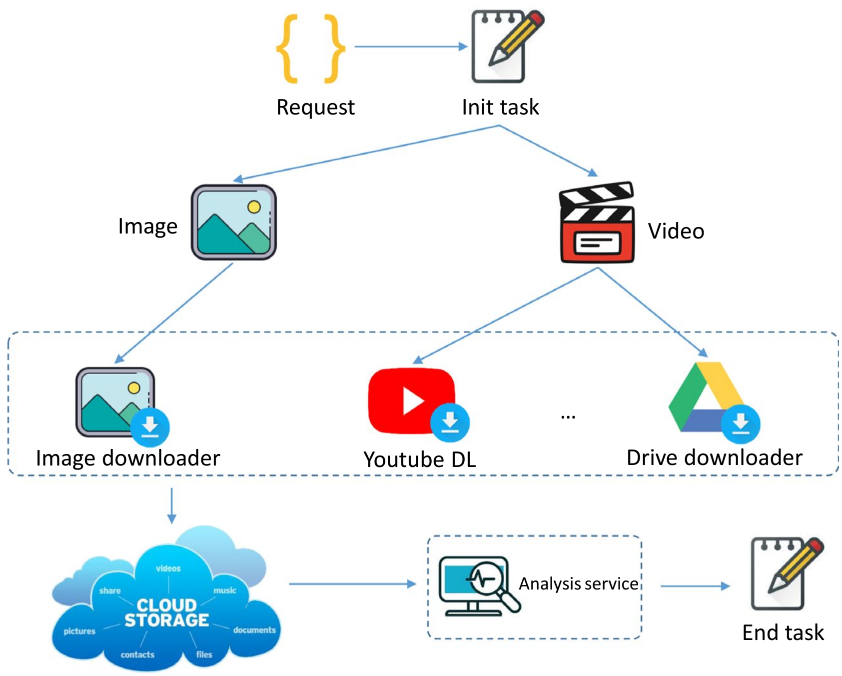


Fig. 11 API procedure of an analysis task for video/image

We have developed and provided the following APIs: sentimental text analysis API, object detection API, face recognition API, and video highlight API.

Case Study

This section presents the system architecture and the main functions of our four prospective systems in potential case studies we have constituted with modules and solutions mentioned in "Proposed architecture for SmartEyes platform".

We aim to demonstrate not only different technologies but also various data types and analysis features in these four case study scenarios. Data should be from a specific context, so it cannot be retrieved directly from specific sources, nor should it be taken directly from that source. Therefore, we developed the services to provide data sources into trusted devices, trusted storage for user provisioning, and authorization.

Face Recognition System

Today, technology connects people's lives everywhere, resulting in the amount of information they are exposed to every day is extremely large. Therefore, people tend to be influenced by reputable people with a high following on social media. Such people influence the community and have the ability to influence the thinking and behavior of a group of people. Therefore, the analysis and evaluation of the popularity of the celebrity's face provide insight understanding to capture social trends and public opinion.

System Overview

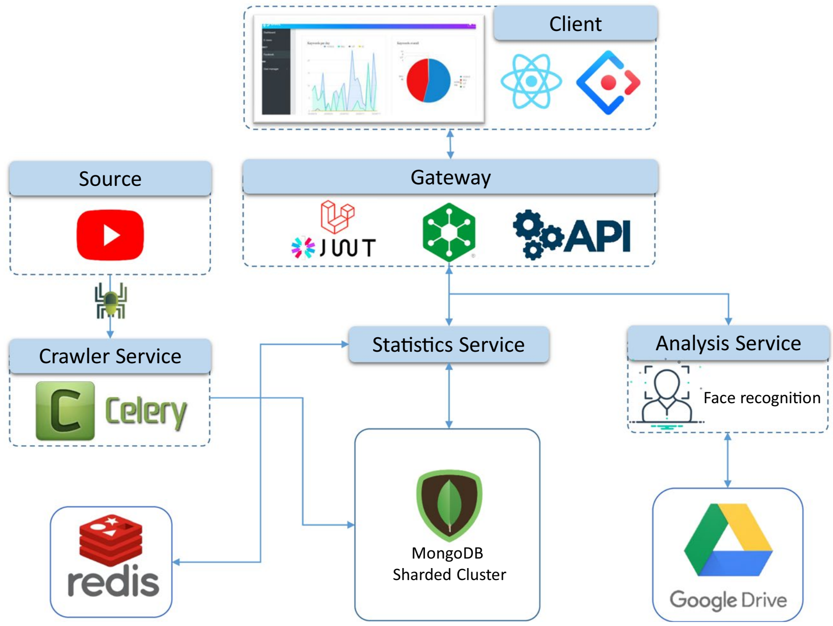
Using our proposed method to custom crawler services and integrate AI service as Face recognition on Google Colab [3] into overview architecture, we have successfully developed a system for collecting, analyzing, and recognizing celebrities' faces in Fig. 12.

The system offers the ability to collect data from YouTube's huge video source, allowing users to choose the channels they want to follow. At the same time, the system helps users track and calculate the statistic of channel parameters from likes, followers to feedback from videos on the channel. In particular, the system supports celebrity facial recognition according to users' needs on videos from the channel they subscribe to. From there, the system counts the number of faces appearances of famous people, then display visually on the analysis charts to help users identify which celebrity appears more on videos to assess the popularity of that person's image on social media.

Figure 12 depicts the system architecture. Specifically, when a user registers for a gateway module account, it processes the request to create a new user. When the user logs in, the JWT on the gateway side generates a token and return it to the client. On subsequent requests, the client attaches that token to the header of the API requests. The gateway module also supports load balancing to ensure a stable distribution of the traffic to the server.

The crawler module is responsible for running periodically to collect data and information on the channels to which the user subscribes. Information includes channel subscriber count, number of likes, number of videos,

Fig. 12 Face recognition system



information, and reactions of each video from that channel. Data after being collected are posted to MongoDB.

Module analysis performs facial recognition from the data requested and saved it to Google Drive. In addition, this module can process the image or video data, which can be from a YouTube link or Google Drive.

API Web Services

The system is supported and syndicated by a set of APIs that track channel information, video, and facial recognition. Following is a list of APIs used in this system:

- Information APIs: get a list of channels, video lists of channels, and details of channels, videos.
- Statistics APIs: get information about subscribers, channel views over time, including start and end, are passed in. Moreover, information about the comments, likes, shares of the video is also supported.
- Face recognition APIs: get into the list of links of videos and images. Then, the system proceeds to download the data according to the link and is identified. Finally, the user makes the API call again to see the job status or result after identification.
- Face model APIs: get links of a dataset and accompanying information of a model. The API then executes the training new model according to the dataset submitted. After the model is trained, users can use it normally.

Statistics and Visualizations

Users can choose to go to the YouTube project to see the comparison statistics between channels. Then, select each channel to see statistics about the number of views and subscribers of that channel. In addition, users can select each channel's video individually to see its own video statistics. These statistics mainly revolve around reactions such as: like, dislike, and comment.

Besides, users can choose the tab Face analysis to see statistics and visualizations on the appearance of the celebrity's face they are tracking in Fig. 13. The image size represents the frequency of the celebrity's appearance on news videos. In addition, the location of such people is linked to predefined coordinates on the map depending on their nationality.

Not only that, but users can also individually select videos to watch the video content and display a list of facial recognition images next to them. The picture automatically moves in sync according to the running video. Next to it is a chart showing the frequency of appearing faces.

Object Detection System

Analysis for products, brand logos, and being mentioned in the text also requires a system to recognize their appearance on images or videos, thereby assessing the popularity, feedback from public opinion in images and videos containing objects being watched. Therefore, the analysis and evaluation of popularity are based on object identification,

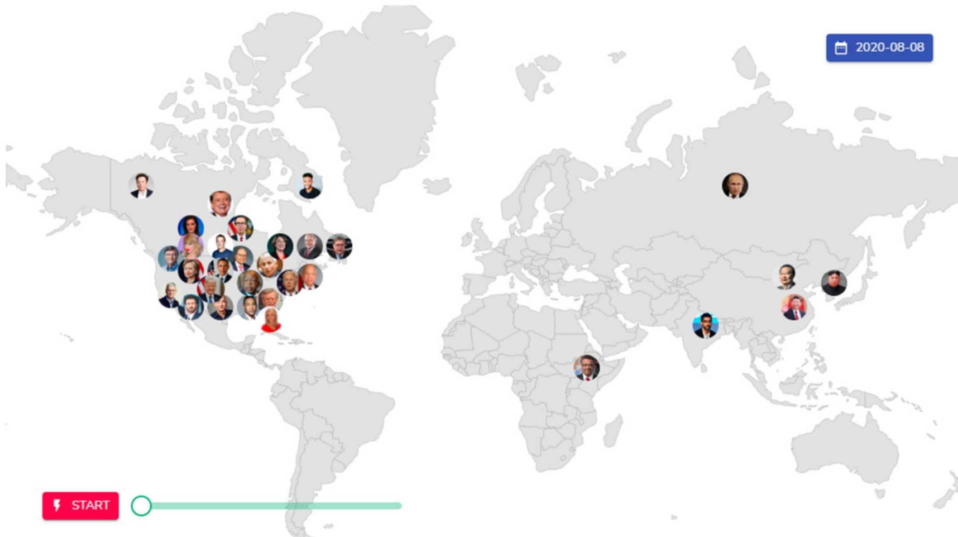


Fig. 13 Maps of celebrities in news videos

specifically for logos, products become essential in capturing and evaluating brand popularity.

System Overview

The system offers similar capabilities to the facial recognition system but it focuses on recognizing objects that the user wants to track its appearance on social networks or other data sources they want. From there, users can grasp which products are outstanding, have a strong influence, and feedback from users for that product.

The object detection system is built based on a common pattern from a general architecture, so there are similarities with the face recognition system mentioned "Face recognition system". The system also has client, gateway, crawler, statistics, and analysis in Fig. 14. The system shows the ability to expand rapidly from the general architectural model, so it can be built quickly just changing module analysis from face recognition to object detection. The tools and libraries used also vary flexibly due to the interoperable architecture of modules. CenterNet is an algorithm applied in this system to identify objects with good results in terms of accuracy and speed.

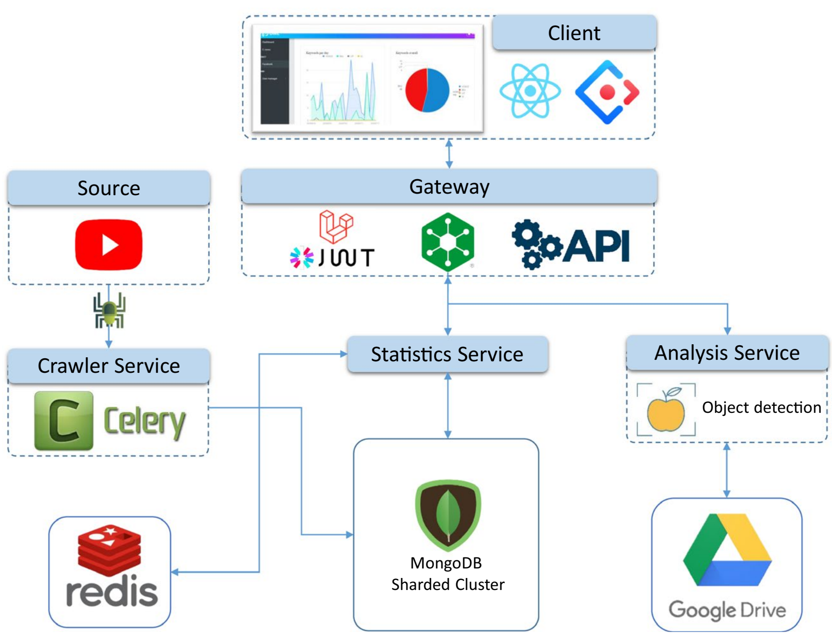


Fig. 14 Object detection system

API Web Services

The system is supported and syndicated by a set of APIs that track channel information, video, and object detection. Following is a list of APIs used in this system:

– Information APIs: get a list of channels, video lists of channels, and details of channels, videos.
– Statistics APIs: get information about subscribers, channel views over time, including start and end, are passed in. Moreover, information about the comments, likes, shares of the video is also supported.
– Object detection APIs: get into the list of links of videos and images. Then, the system proceeds to download the data according to the link and is identified. Finally, the user makes the API call again to see the job status or result after identification. We use CenterNet to implement the main processing of these APÍ and Object model APIs below.
– Object model APIs: Get links of a dataset and accompanying information of a model. The API then executes the training new model according to the dataset submitted. After the model is trained, users can use it normally.

Statistics and Visualizations

The statistical and visualization features of this system are similar to those in the Face recognition system. However, instead of focusing on celebrities' faces, we focus on object detection, brand logos. Specifically, we tested the system to recognize the logo of the Compact drinking water brand in the TV show—the Brain Vietnam in Fig. 15.

Video Highlight System

With the bustling pace of people in the information technology age, daily news is always updated quickly. The request is how to briefly summarize highlights from a very long source so that people can easily grasp them when they have time to follow. A good example is football matches that range from 90 to 120 min. For football fans, it is difficult to ignore good matches, but there are also cases where they can only see the summaries of the matches due to limited time. Because of this, video data processing and outstanding happenings summary become essential in such cases.

System Overview

Thanks to combining the overall architecture with the right tools and solutions, we have successfully built a system to collect, analyze, and summarize the outstanding progress of football matches. The system offers the ability to collect video data from football channels from social networks such as YouTube and supports statistics of match videos from the football channel. The most prominent feature is the match video summary support to become a short featured video to help users grasp the main action of the match quickly.

The system is still built on the overview architecture such as the face recognition system as well as the object detection system. The system is still connected to a client, gateway, crawler, statistics, and analysis in Fig. 16. The system changes are mainly in module analysis and client-side interface.

Module analysis receives the required video data, then processes and summarizes video evolution based on the audio transformation throughout the video. The data after

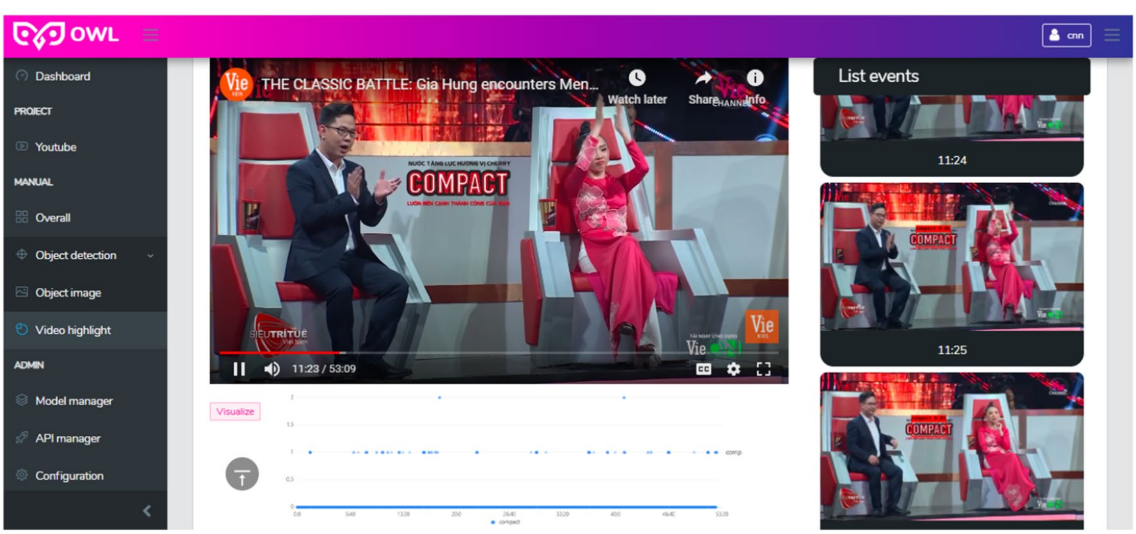


Fig. 15 Object detection for the Compact brand in a video

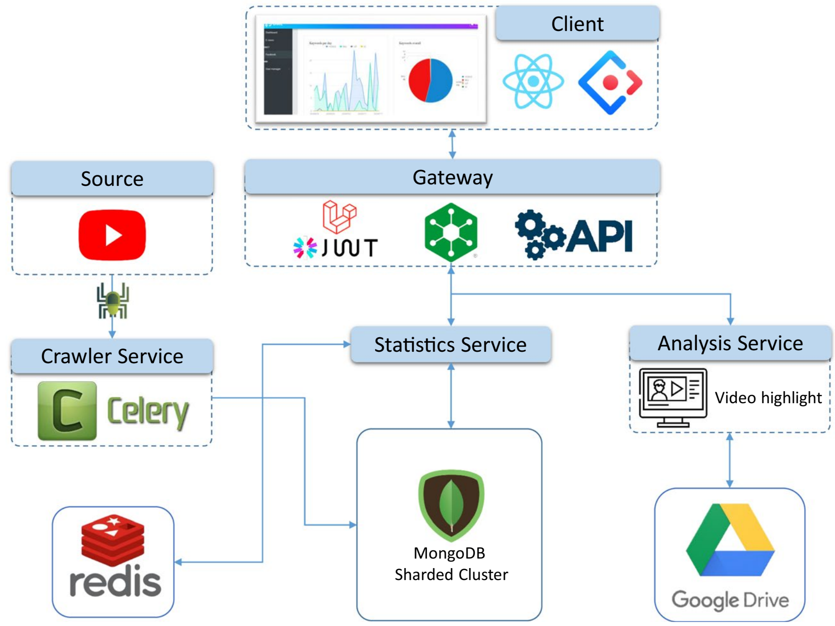


Fig. 16 Video highlight system

the summary are stored on Cloud storage like Google Drive and are made available to the client upon request.

API Web Services

The system is supported and syndicated by a set of APIs that track channel information, video, and highlight video. Following is a list of APIs used in this system:

- Information APIs: get a list of channels, video lists of channels, and details of channels, videos.
- Statistics APIs: get information about subscribers, channel views over time, including start and end, are passed in. Moreover, information about the comments, likes, shares of the video is also supported.
- Video highlight APIs: get into the list of links of videos. Then, the system proceeds to download the data according to the link and is identified. Finally, the user makes the API call again to see the job status or result after identification. We use the technique of video splitting and clustering high volume clips to summarize the video, which is covered in "Video highlight".

Text Analysis System

Every time the enrollment season comes, universities less or more implement communication strategies to attract students or help high school students grasp the enrollment situation to make appropriate decisions in choosing a university and major. Therefore, the need to analyze which schools and faculties are mentioned a lot, have high shares, and responses to capture the effectiveness in universities' communication becomes necessary.

System Overview

We have built a system of collecting, analyzing, and statistics keywords appearing on posts on social networks. The system offers the ability to collect data from social networks, namely Facebook, to capture information of admission pages, then statistics and track shares, and likes to capture the situation of daily admission information.

The system is again built on the general architecture that we used to develop previous systems. The system still includes the client, gateway, crawler, statistics, and analysis in Fig. 17.

The change mainly lies in the crawler module when the system mainly collects text data from the world's most popular social network, Facebook. In addition, the system also supports electronic newspapers such as VnExpress—the most viewed Vietnamese newspaper.

API Web Services

The system is supported and syndicated by a set of APIs that track fan pages information from Facebook with their post, posts from VnExpress, and text analysis with counting keywords and sentimental analysis. Following is the list of APIs used in this system:

- Information APIs: get a list of fan pages, post lists of fan pages, and details of fan pages, posts.

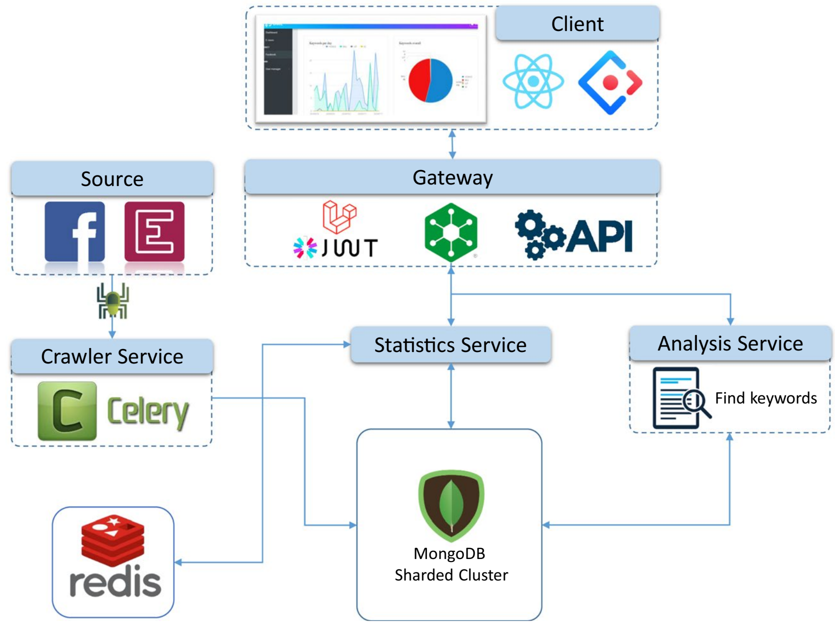


Fig. 17 Text analysis system

- Statistics APIs: get information about likes, shares over time, including start and end, which are passed in. Moreover, information about the likes, shares of the post is also supported. We use Facebook Scraper tool to get that information.
- E-news APIs: get posts from a website for news on the Internet. We use these APIs to get posts from the most viewed Vietnamese newspaper like VnExpress. We can apply counting keywords APIs to analyze the frequency of appearing on keywords of interest.
- Counting keywords APIs: the input text data are standardized again by bringing all the characters to lowercase and removing the accent marks. Then, use the dictionary provided to count the number of keyword occurrences.
- Sentimental analysis APIs: we used pre-trained sentiment analysis models for English, after processing textual data, they are classified into one of two categories: positive and negative. We also study to use PhoBERT for getting the highest speed and accuracy possible on Vietnamese.

Statistics and Visualizations

We use the virtualized list of React [24] to optimize performance when displaying a large list of posts. Besides, the statistical function is intuitive through the charts of the AntV G2 Plot Chart library. Users can choose the news viewing function with articles from the VnExpress source summarized and go to the original link to view details. In addition, there are statistical functions that compare the shares and likes of enrollment pages visually. Besides, showing the statistics of an admission page with familiar parameters of likes and shares. Users can view the posts gathered from the enrollment pages they follow. The function of tracking keyword appearance is statistically and visually displayed in the Text analysis tab in Fig. 18.

We use the StackArea chart to visualize the frequency of appearing related keywords of the universities mentioned on the admission fan pages. Specifically, the number of keyword occurrences in a day during a month. Next to that is a pie chart that shows the total number of occurrences during that time.

Conclusion

In this paper, we present in detail about SmartEyes, our flexible platform to support creating systems to collect and analyze social media data. We first introduced the preliminary implementation of SmartEyes in [15]. In SmartEyes, we emphasize the flexible architecture so that new components and features can be integrated easily. Furthermore, we provide components and utilities used to quickly create open data providers based on different social media sources.

Our architecture for SmartEyes consists of four main components: gateway, crawler, statistics, and analysis. We link modules and services through APIs and centrally managed data storage. Diverse data sources are text data stored on the database and video/images stored on Cloud storage. Besides, our method focuses on the ability to flexibly integrate artificial intelligence analysis functions into the system to analyze social events in an intelligent way and serve smart cities in creating open data providers. Moreover, using artificial intelligence functions as a replaceable module according to user needs will bring extremely high scalability.

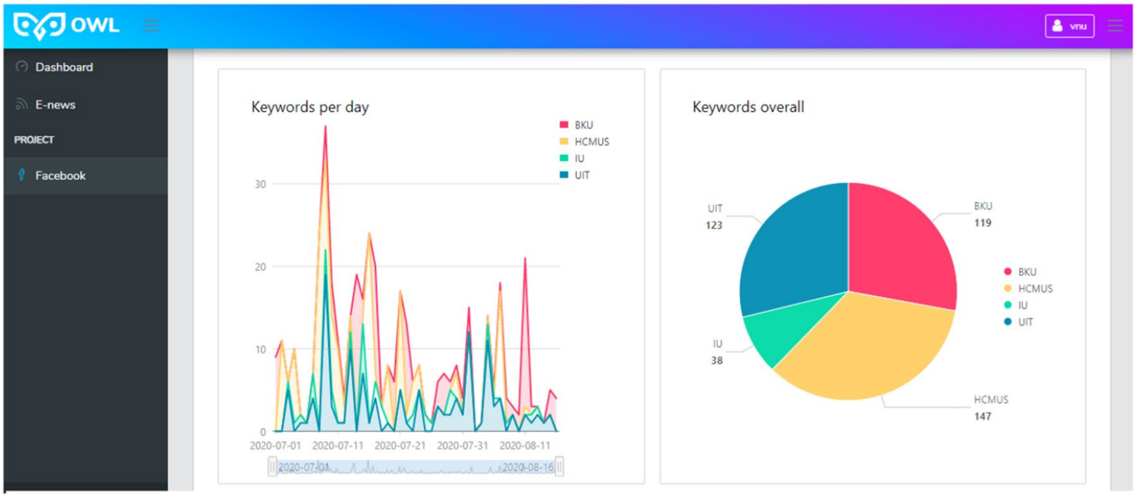


Fig. 18 Keywords of universities mentioned on Facebook pages

In our experiments for the four demonstration applications, we deploy services related to data analysis with artificial intelligence on Google Colab [3]. Services can be deployed on other similar server services. In addition, the centralized data stored on Cloud storage gives the advantage of handling private and user-provided data cases such as surveillance cameras, personal data, or organizations' data.

Using our SmartEyes platform, we expect to create various social multimedia analysis applications quickly and efficiently. Therefore, we developed four systems to illustrate our proposed solutions. The four systems include a facial recognition system for celebrity recognition, an object detection system for brand logo recognition, a video bookmarking system for football match summaries, and a text analysis system for keyword occurrence and emotional text analysis for college admissions. Each system represents a specific potential application in society. SmartEyes can also be used to create open data providers efficiently to support the development of smart cities and smart environments.

Currently, we are studying other text analyses for Vietnamese with PhoBERT [19], researching for other artificial intelligence to integrate into our systems. We will also research new mechanisms and solutions to increase system performance, load balancing, data visualization, and security. We aim to quickly and flexibly integrate solutions to support open data providers in smart cities now and in the future.

Funding Thanh-Cong Le was funded by Vingroup Joint Stock Company and supported by the Domestic Master/PhD Scholarship Programme of Vingroup Innovation Foundation (VINIF), Vingroup Big Data Institute (VINBIGDATA), code VINIF.2020.ThS.JVN.05. This work was also funded by Gia Lam Urban Development and Investment Company Limited, Vingroup, and partially supported by Vingroup Innovation Foundation (VINIF) under project code VINIF.2019.DA19.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Adarsh P, Rathi P, Kumar M. Yolo v3-tiny: object detection and recognition using one stage improved model. In: 2020 6th international conference on advanced computing and communication systems (ICACCS). 2020. p. 687–94.
2. Ahmed S, Mahmood Q. An authentication based scheme for applications using JSON web token. In: 2019 22nd international multitopic conference (INMIC). 2019. p. 1–6.
3. Bisong E. Google colaboratory. Berkeley: Apress; 2019. p. 59–64. https://doi.org/10.1007/978-1-4842-4470-8_7.
4. Bogdan Batrinca PCT. Social media analytics: a survey of techniques, tools and platforms. AI Soc. 2015;323(6088):89–116.
5. Boyko N, Basystiuk O, Shakhovska N. Performance evaluation and comparison of software for face recognition, based on dlib and opencv library. In: 2018 IEEE second international conference on data stream mining processing (DSMP). 2018. p. 478–82.
6. Chen X, Zhang Q, Han J, Han X, Liu Y, Fang Y. Object detection of optical remote sensing image based on improved faster RCNN. In: 2019 IEEE 5th international conference on computer and communications (ICCC). 2019. p. 1787–91.
7. Chi X, Liu B, Niu Q, Wu Q. Web load balance and cache optimization design based nginx under high-concurrency environment. In: 2012 third international conference on digital manufacturing automation. 2012. p. 1029–32.

8. Clayton H, Gilbert E. Vader: a parsimonious rule-based model for sentiment analysis of social media text. Behav Res. 2014;323(6088):1–8.
9. Dong Yu LD. Automatic speech recognition. In: Automatic speech recognition. A deep learning approach. New York: Springer; 2015. p. 1–7.
10. Elizabeth MM, Chareen S, Elison-Bowers P. Image and video disclosure of substance use on social media websites. Comput Hum Behav. 2010;26(6088):1405–11.
11. free A. open resource with Machine Learning papers, c., evaluation tables: Paper With Code. 2020. https://paperswithcode.com/.
12. Geitgey A. Modern face recognition with deep learning. 2018. https://github.com/ageitgey/face_recognition.
13. Hearst MA. Support vector machines. IEEE Intell Syst. 1998;13(4):18–28. https://doi.org/10.1109/5254.708428.
14. Kim HM, Sabri S, Kentc A. Smart cities for technological and social innovation case studies, current trends, and future steps. Amsterdam: Elsevier; 2021.
15. Le TC, Nguyen QV, Tran MT. Flexible platform for integration, collection, and analysis of social media for open data providers in smart cities. In: Dang TK, Küng J, Takizawa M, Chung TM, editors. Future data and security engineering. Cham: Springer International Publishing; 2020. p. 304–24.
16. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL. Microsoft coco: common objects in context. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. Computer Vision—ECCV 2014. Cham: Springer International Publishing; 2014. p. 740–55.
17. Liu H. Highlight extraction in soccer videos by using multimodal analysis. In: 2017 13th international conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD). 2017. p. 2169–73.
18. Min C, Shiwen M, Yunhao L. Big data: a survey. Mobile Netw Appl. 2014;323(6088):171–209.
19. Nguyen DQ, Nguyen AT. Phobert: pre-trained language models for Vietnamese. 2020.
20. Ren H, Li Z. Object detection using edge histogram of oriented gradient. In: 2014 IEEE international conference on image processing (ICIP). 2014. p. 4057–61.
21. Scott AC, Kristopher K, Danielle SM. Sentiment analysis and social cognition engine (seance): an automatic tool for sentiment, social cognition, and social-order analysis. Behav Res. 2016;323(6088):804–6.
22. Sharma S, Shanmugasundaram K, Ramasamy SK. Farec–cnn based efficient face recognition technique using dlib. In: 2016 international conference on advanced communication control and computing technologies (ICACCCT). 2016. p. 192–95.
23. Shunji M, Nishida H, Yamada H. Optical character recognition. New York: Wiley; 1999. p. 1–7.
24. Vaughn B. Virtualized list. 2020. https://bvaughn.github.io/react-virtualized.
25. Wild Home LF. A public benchmark for face verification. 2020. http://vis-www.cs.umass.edu/lfw.
26. William I, Ignatius Moses Setiadi DR, Rachmawanto EH, Santoso HA, Sari CA. Face recognition using facenet (survey, performance test, and comparison). In: 2019 fourth international conference on informatics and computing (ICIC). 2019. p. 1–6.
27. Womg A, Shafiee MJ, Li F, Chwyl B. Tiny ssd: a tiny single-shot detection deep convolutional neural network for real-time embedded object detection. In: 2018 15th conference on computer and robot vision (CRV). 2018. p. 95–101.
28. Zeng H, Zhang Z, Shi L. Research and implementation of video codec based on ffmpeg. In: 2016 international conference on network and information systems for computers (ICNISC). 2016. p. 184–88.
29. Zhou X, Wang D, Krähenbühl P. Objects as points. 2019. arXiv:1904.07850 [CoRR abs].
30. Zulko. Automatic soccer highlights. 2014. http://zulko.github.io/blog/2014/07/04/automatic-soccer-highlights-compilations-with-python.