YouTube Trending Video Analysis using AWS ECS and Elasticsearch

Ashu Kumar
Concordia University
ashukumartgk@gmail.com

Vishnu Rameshbabu
Concordia University
vishnuramesh200@gmail.com

Ayyanar Kadalkani
Concordia University
aayanmasters@gmail.com

Benjamin Douglas Joseph David
Concordia University
douglasj0811@gmail.com

Hani Saravanan
Concordia University
hanisaravanan@gmail.com

Jothi Basu LKV,
Concordia University
jothibasulkv01@gmail.com

ABSTRACT

In this project, we deploy a sophisticated architecture for YouTube Trending Video data analysis, utilizing AWS Elastic Container Service (ECS) with autoscaling and Spot Fleet, alongside Elasticsearch. Our workflow begins with a custom Docker image pushed to ECS, where it runs on an EC2 instance. This EC2 instance benefits from the cost-efficiency and scalability of Spot Fleet, combined with the robustness of ECS autoscaling, ensuring optimal resource utilization. The Docker container executes the master code, which dynamically selects one of three analyses based on input from the YouTube Trending Video dataset stored in Elasticsearch. This setup allows for efficient data management and retrieval. Each analysis is meticulously designed to extract specific insights, such as trend patterns and viewer engagement. After analysis, the findings are stored back in Elasticsearch for enhanced access and visualization. This architecture, integrating advanced AWS services and Elasticsearch, exemplifies our commitment to leveraging cutting-edge technology for efficient, scalable, and insightful analysis of digital media trends.

1 INTRODUCTION

Our project embarks on a granular exploration of YouTube's trending phenomena, harnessing the power of data to uncover the intricacies of video popularity. We dissect the temporal and geographical dimensions of YouTube viewership, focusing on the performance of channels on a monthly basis across different countries and years. This nuanced approach allows us to reveal which YouTube channels dominate the landscape, with a specific emphasis on view count as the primary indicator of trendiness.

To facilitate this deep-dive analysis, we employ an advanced architecture within Amazon Web Services (AWS). The journey begins with our Docker image, a bespoke analytical tool, being pushed into AWS Elastic Container Service (ECS). Here, the ECS orchestrates the running of this image on an EC2 instance, a virtual engine room where our analytical processes come to life.

Inside the EC2, the Docker container executes the master code, which intelligently selects from three analytical strategies depending on the dataset at hand. The first strategy is the month-over-month, year-over-year trending analysis by country, providing insights into the most viewed channels within specified temporal windows. The second strategy examines global viewer preferences across different video categories, determining the most and least favored content types. The third and final strategy investigates the presence of universally trending videos, identifying content that resonates across different national audiences.

Data for these analyses is sourced and stored in Elasticsearch, which serves as both the input repository and the final destination for our insights. This seamless integration allows for real-time data processing and immediate retrieval of results. After the analysis is complete, the enriched data is stored back into Elasticsearch indices, laying the groundwork for our next phase.

Using Elasticsearch Kibana, we transform our analytical findings into compelling visualizations. These visual aids are not mere representations of data but are narrative tools that illustrate the ebb and flow of YouTube's trending videos, providing actionable insights into viewer engagement and content strategy. In sum, our project is not only a technical endeavor but also a storytelling exercise, where data narrates the evolving tale of digital content trends.

2 SYSTEM ARCHITECTURE

The system's architecture is designed to leverage Amazon Web Services (AWS) for robust, fault-tolerant, and scalable analysis of YouTube Trending Video data. Figure 1 represents the architecture. This architecture ensures high availability and efficient distribution of workload across services.

2.1 AWS Elastic Container Service (ECS)

AWS ECS serves as the orchestration service for our containerized applications. It manages the deployment, scaling, and administration of the application containers across a cluster of EC2 instances. ECS provides fault tolerance by monitoring the health of applications and automatically replacing unhealthy instances, thereby maintaining the desired count of active containers.

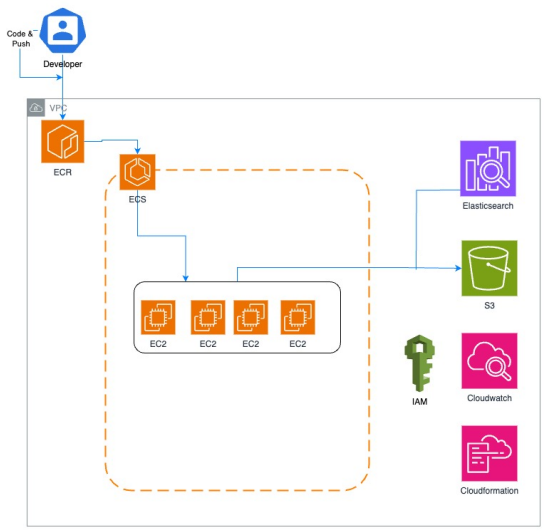

Figure 1: System Architecture

2.2 Amazon Elastic Container Registry (ECR)

ECR works in conjunction with ECS, providing a secure repository to store and manage our Docker images. It integrates with IAM for resource-level permissions, ensuring secure image distribution and fault tolerance against unauthorized access.

2.3 Amazon EC2

EC2 instances, provisioned through ECS, offer scalable computing capacity. EC2 allows us to utilize Spot Instances for cost savings and Auto Scaling to handle variable workloads, ensuring that our system scales efficiently with the fluctuating demands of data processing tasks. Spot Fleet Management further enhances fault tolerance by diversifying the risk across various spot markets.

2.4 Elasticsearch

Elasticsearch offers a distributed search engine, capable of scaling horizontally to accommodate the growing dataset sizes while providing quick data retrieval. It adds to the fault tolerance of the system by replicating data across multiple nodes, ensuring no single point of failure. Elasticsearch's built-in sharding capabilities ensure distributed data storage and parallel processing, enhancing query performance and system reliability.

Overall, this architecture is designed to ensure that our analytical system can handle large volumes of data with consistent performance, automatically adjust to the incoming data load, and maintain uninterrupted operation, even in the face of component failures.

3 PREPROCESSING DATASET

Utilizing Python and the Pandas library, we read the initial dataset from a CSV file and employed a 'columns' parameter to eliminate irrelevant columns, streamlining the data for precise analysis. We included separate column named country_name to record each country names.This pre-processing was executed iteratively for 11 distinct countries, amalgamating the results into a singular, cleansed file named 'preprocessed_single_file_dataset.csv', resulting in a comprehensive 1.2 GB CSV file. This preparatory step was critical in ensuring that the subsequent analyses were conducted on accurate and pertinent data, setting a solid foundation for reliable insights.

4 WORKFLOW

Our system's architecture, as depicted in Figure 1, is crafted to process a large volume of documents efficiently and reliably over several years, from 2020 to 2023. The architecture is designed to handle multiple versions of analysis for each year, ensuring that the system remains both flexible and robust.

The workflow begins with the developers writing code and pushing it to the Amazon Elastic Container Registry (ECR) within a Docker environment. This containerized code, once deployed to ECR, is then pulled into the AWS Elastic Container Service (ECS), where a service is created. ECS manages the deployment and dynamically scales the number of EC2 instances as needed, from a few to potentially hundreds, to accommodate the processing demand.

These instances pull the required JSON data from Amazon S3, which acts as a storage service for the data needed by the code. If any file is missing, the master code—running on each EC2 instance—retrieves it. The master code (master.py) along with three distinct analysis scripts then interact with Elasticsearch to check for new data to be processed.

Once new data is identified, the master node initiates the processing of the relevant analysis. It ensures that once a dataset is under processing, its status is updated from new to processing, to avoid duplicate processing by other nodes. This is part of a fault tolerance mechanism where, if the processing timestamp exceeds five minutes—a sign that the node might have crashed—the status is evaluated, and corrective action is taken.

Additional AWS services integrated into this architecture include Identity and Access Management (IAM) for secure access control, CloudWatch for monitoring and debugging, and CloudFormation for quick deployment of the infrastructure. These services are instrumental in maintaining the system's security, monitoring, and scalability. This architecture ensures that our deployment is not only efficient in processing but also robust against failures, thereby minimizing downtime and maintaining data integrity.

5 VISUALIZATION

The visualization component of our report is executed through the utilization of Kibana, a potent data visualization tool that interfaces seamlessly with Elasticsearch. Within Kibana, we have crafted three distinct dashboards, each corresponding to one of the analyses conducted in our project. These dashboards serve as the visual synthesis of our data-driven insights, translating complex datasets into intuitive and interactive graphical representations.

Each dashboard is meticulously designed to encapsulate the essence of the respective analysis it represents. The first dashboard visualizes the month-over-month, year-over-year trending channels by country, leveraging bar charts to display the channels' view counts comparatively. Figure 2 shows the bar graph for Canada Trending Channels. This allows for the identification of persistent trends as well as the emerging popularity of channels over time.

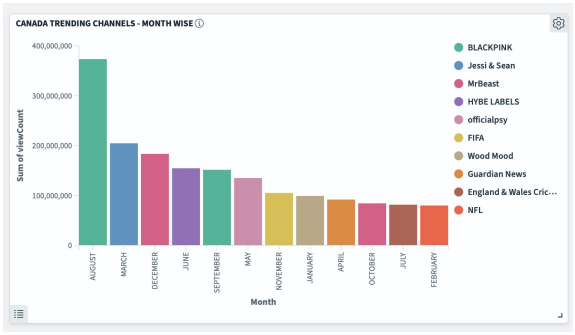

Figure 2: Canada Trending Channels

The second dashboard focuses on the global preferences in video categories. Through its visual elements, we can discern the most and least favored content types across the globe. The use of bar charts enables a straightforward comparison of categories, highlighting the maximum and minimum engagement metrics for each.

The third dashboard delves into the commonalities of trending videos across various countries. It provides a visual breakdown of videos that have achieved international popularity, offering insights into the universal appeal of certain content.

Our dashboards not only offer a high-level overview of key metrics but also simplify data exploration, allowing us to drill down into specific aspects as needed. The graphical representations in Kibana facilitate a more accessible understanding of complex data, making it possible to grasp subtle nuances of digital video trends at a glance.

6 CONCLUSION

In conclusion, our project has established a highly efficient and resilient system for processing YouTube Trending Video data, as illustrated in Figure 1. This system has been rigorously designed to manage and analyze large volumes of documents spanning multiple years, demonstrating remarkable flexibility and robustness. By leveraging AWS services, we have ensured that our architecture is not only scalable but also maintains data integrity and fault tolerance.

The successful deployment of this architecture has resulted in a system that dynamically adapts to processing loads and guards against potential failures. This ensures that our analyses remain current and accurate, reflecting the evolving trends within YouTube's vast content landscape. The integration of AWS components like ECR, ECS, EC2, S3, IAM, CloudWatch, and CloudFormation has been instrumental in achieving a seamless workflow from code deployment to data analysis.

References

[1] Amazon Web Services (2020). *Amazon ECS - Run containerized applications in production.* Retrieved from `https://aws.amazon.com/ecs/`.

[2] S. Ifrah (2019). *Scaling the AWS EKS, ECS, and ECR Containerized Environments.* In *Deploy Containers on AWS* (pp. 255–288). Apress. DOI: 10.1007/978-1-4842-5101-0_7.

[3] N. Shah, D. Willick, & V. Mago (2022). *A framework for social media data analytics using Elasticsearch and Kibana.* Wireless Networks, 28(3), 1179–1187. DOI: 10.1007/s11276-018-01896-2.

[4] P. M. Dhulavvagol, V. H. Bhajantri, & S. G. Totad (2020). *Performance Analysis of Distributed Processing System using Shard Selection Techniques on Elasticsearch.* In *Procedia Computer Science,* Vol. 167, pp. 1626–1635. Elsevier B.V. DOI: 10.1016/j.procs.2020.03.373.

[5] AWS (2021). *Was ist Amazon Elastic Container Service?* Retrieved from `https://docs.aws.amazon.com/de_de/AmazonECS/latest/developerguide/Welcome.html`.

[6] B. McLaughlin & S. Perrott (2020). *CloudFormation.* In *AWS Certified SysOps Administrator Study Guide, 2E* (pp. 381–400). Wiley. DOI: 10.1002/9781119561569.ch18.

[7] Mr. D. Prakash (2023). *CAMPUS CLOUD: EMPOWERING UNIVERSITY MANAGEMENT WEB APPLICATION WITH CLOUD-HOSTED DOCKER TECHNOLOGY ON AWS.* INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT, 07(04). DOI: 10.55041/ijsrem20366.

[8] W. Sholihah, S. Pripambudi, & A. Mardiyono (2020). *Log Event Management Server Menggunakan Elastic Search Logstash Kibana (ELK Stack).* JTIM: Jurnal Teknologi Informasi Dan Multimedia, 2(1), 12–20. DOI: 10.35746/jtim.v2i1.79.

[9] AWS (2021). *Amazon CloudWatch - Application and Infrastructure Monitoring.* Retrieved from `https://aws.amazon.com/cloudwatch/`.

[10] C. Tarigan, V. J. L. Engel, & D. Angela (2018). *Sistem Pengawasan Kinerja Jaringan Server Web Apache dengan Log Management System ELK (Elasticsearch, Logstash, Kibana).* Jurnal Telematika Edisi Industrial Engineering Seminar and Call for Paper (IESC), 7–14.