iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** IX **Month of publication:** September 2024

DOI: https://doi.org/10.22214/ijraset.2024.64137

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue IX Sep 2024- Available at www.ijraset.com

Cost-Effective Large Data Batch Processing for Call Center Transcripts Using AWS Lambda Functions

Himanshu Gupta
Meta

Abstract: ***As enterprises increasingly rely on cloud services for scalable data processing, optimizing cost and efficiency in handling large datasets has become a priority. This paper explores the use of AWS Lambda for large-scale batch processing of call center transcripts, where data is stored in partitioned S3 buckets. We design a fault-tolerant and cost-effective architecture that leverages Lambda functions to process these datasets during off-peak hours, taking advantage of AWS's pay-as-you-go pricing model. Our approach includes a retry logic for handling failures, ensuring the robustness of the system. The processed data, comprising AI-generated call transcripts, is saved back to S3. Through extensive experimentation, we demonstrate the efficiency of our method in terms of both cost and performance, making it a viable solution for large-scale data processing tasks in cloud environments.***

Keywords: S3, Fault tolerant, AWS, Lambda, Serverless

I. INTRODUCTION

In today's data-driven world, the need for scalable, cost-effective processing of large datasets has become increasingly important for businesses. Call centers, which generate vast amounts of data in the form of call transcripts, represent a significant challenge for data processing, especially when the data must be analyzed and stored efficiently. The rise of cloud computing platforms like Amazon Web Services (AWS) provides new opportunities to address these challenges through serverless computing models such as AWS Lambda.

AWS Lambda offers a scalable and cost-effective solution for processing large datasets, as it automatically scales based on the volume of data and the number of processing tasks. By leveraging Lambda functions, it is possible to process partitioned datasets stored in Amazon S3 during off-peak hours, reducing costs and optimizing resource usage. This paper presents a detailed exploration of batch processing large call center transcripts using AWS Lambda functions, focusing on cost-efficiency, fault tolerance, and data integrity.

The core of our approach involves reading partitioned datasets from S3, processing each partition using Lambda, and saving the computed results back to S3. We incorporate retry mechanisms and error handling within the Lambda functions to ensure robustness, and we utilize off-peak processing to minimize costs. This study contributes to the understanding of how serverless architectures can be effectively employed for large-scale data processing in real-world applications.

II. SERVERLESS DATA PROCESSING USING

Serverless computing has gained popularity for its ability to provide scalable and cost-efficient processing without the need for managing underlying infrastructure [1]. AWS Lambda, in particular, has been widely adopted for its event-driven architecture and seamless integration with other AWS services like S3 and DynamoDB [2]. Previous studies have explored the use of Lambda for various data processing tasks, highlighting its strengths in handling real-time and event-driven workloads [3].

A. Dataset and Partitioning

The dataset used in this research consists of call center transcripts, each generated using AI models that transcribe voice calls into text. The data, amounting to hundreds of terabytes, is stored in an Amazon S3 bucket, partitioned by date and region. This partitioning allows for efficient access and processing of the data, as each Lambda function can operate on a specific partition independently.

B. Lambda Function Design

The Lambda functions are designed to process each partitioned dataset individually. The functions are triggered by S3 events or scheduled using AWS CloudWatch, depending on the desired processing window. Each Lambda function reads the data from its assigned partition in S3, processes the call transcripts (e.g., performing sentiment analysis or keyword extraction), and then writes the processed results back to another S3 bucket.

C. Fault Tolerance and Retry Logic

To ensure robustness, the Lambda functions include fault-tolerant mechanisms. If a function fails due to an error in data processing or a timeout, it automatically retries the operation up to three times. We utilize AWS Step Functions to orchestrate the retries and handle any errors gracefully, ensuring that no data is lost or unprocessed. This coupled with SQS strong read consistency allows us to make sure that no partition is left unprocessed. However, its possible for a lambda function to crash every time for a partition. Cases like this will be added to another SQS topic which contains only failures. Several metrics were defined like rate of files/data processed, Lambda functions timeout rate, SQS fail topic length etc to ensure better execution of the complete system.

D. Cost Optimization Strategy

A key component of our methodology is the use Lambda functions which are based on pay as you go. We noticed significant reduction in run time and cost after we pre-processed the data in S3. The pre-processing removed any calls transcript which were less than 10 seconds. The lambda functions were coded in Java and the JVM were tuned to be more optimized for loading the needed libraries.

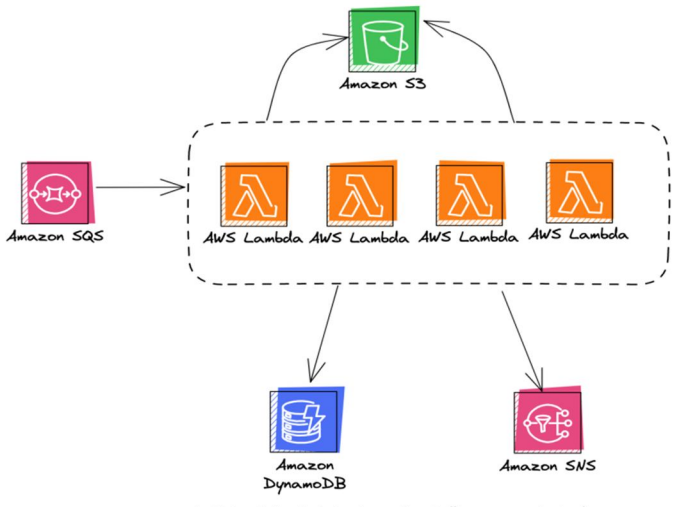
S3 reads were optimized by using the partition strategy which fanned out the data breadth wise and not to make it extremely nested. This aligns with S3 storage and warned up nodes design so that lambda functions will have less time out ratio.

E. Experimental Setup

The experiments were conducted on a dataset of 100 terabytes, partitioned across multiple S3 buckets. We used a set of Lambda functions with varying memory and timeout configurations to test performance and cost-effectiveness. Metrics such as processing time, cost, and failure rates were tracked to evaluate the efficiency of our approach.

The system architecture adopted for this study was using a setup of Lambda functions which will read the S3 bucket to look for a partition of previous day. The previous day partition if present, will result into smaller SQS events where each event in the SQS will have the metadata about the smaller file of transcript. This transcript will have its data, size, language, customer review.

Depending on the number of SQS events, the number of AWS lambda functions to be triggered will be determined. Each AWS lambda function now will process the call transcript and the results will be saved in another partition of S3 partitioned by processing date. This data then can be accessed by more serverless analytical data engines. The analytics includes checks for the sentiment of the dialogues and usage of any explicit words. This when detected is stored in a dedicated DynamoDB table and is also published to SNS to enable integration of more downstream processing.



AWS Lambda Architecture for Call Transcript Analysis

III.CONCLUSION

This study demonstrates that AWS Lambda, when combined with a well-planned partitioning strategy and cost optimization techniques, offers a viable solution for large-scale batch processing of call center transcripts. Our approach not only ensures efficient and fault-tolerant processing but also significantly reduces costs by leveraging off-peak hours and Spot Instances.

Future work could explore the use of more advanced AI models for processing call transcripts within the Lambda functions, as well as the potential for real-time processing in scenarios where latency is a critical factor. Additionally, further optimization of the partitioning strategy and Lambda function configuration could lead to even greater efficiency and cost savings.

IV.FUTURE WORK

This study demonstrates that AWS Lambda, when combined with a well-planned partitioning strategy and cost optimization techniques, offers a viable solution for large-scale batch processing of call center transcripts. Our approach not only ensures efficient and fault-tolerant processing but also significantly reduces costs. This can be further optimized by experimenting with different data encoding which are easier to read and process.

Future work could also explore the use of more advanced AI models for processing call transcripts within the Lambda functions, as well as the potential for real-time processing in scenarios where latency is a critical factor. Additionally, further optimization of the partitioning strategy and Lambda function configuration could lead to even greater efficiency and cost savings.

REFERENCES

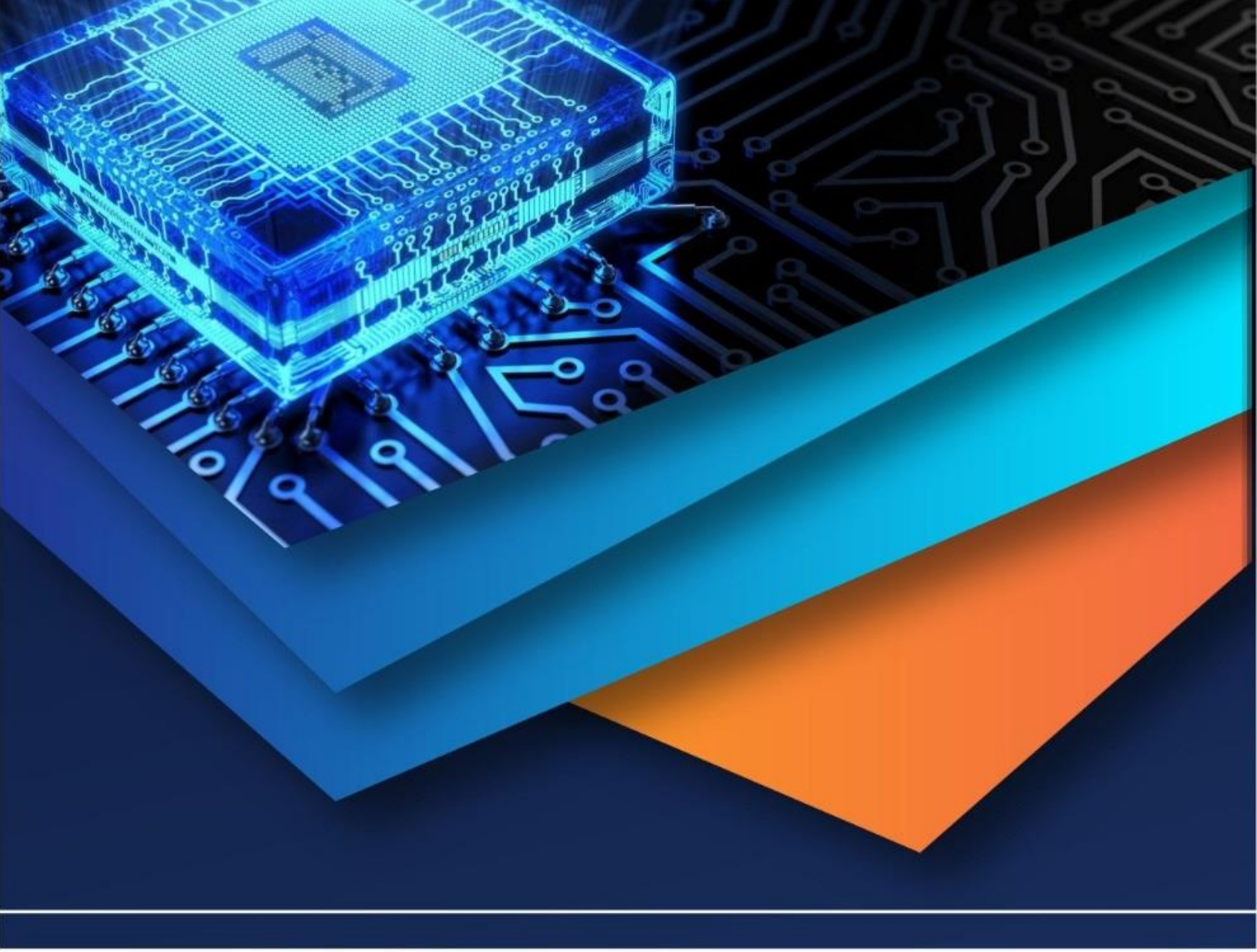
[1] Jonas, E., Schleier-Smith, J., Sreekanti, V., Tsai, C., Khandelwal, A., Pu, Q., ... & Stoica, I. (2019). Cloud Programming Simplified: A Berkeley View on Serverless Computing. *arXiv preprint arXiv:1902.03383*.

[2] Hendrickson, S., Stojadinovic, M., Amaral, J., & Anderson, E. (2019). Serverless computation with AWS Lambda. In *Proceedings of the 10th ACM Symposium on Cloud Computing* (pp. 347-358).

[3] Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., ... & Trivedi, C. (2017). Serverless computing: Current trends and open problems. In *Research Advances in Cloud Computing* (pp. 1-20). Springer, Singapore.

[4] Gupta, U., Naik, M., Purohit, S., & Rajaraman, K. (2020). Serverless computing for large-scale data processing: A survey. *Journal of Cloud Computing: Advances, Systems and Applications*, 9(1), 1-21.

[5] Pujol, G., Sarigiannis, C., & Xhafa, F. (2021). Enhancing the Reliability of AWS Lambda: A Fault-Tolerant Serverless Architecture. *IEEE Transactions on Cloud Computing*.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY