INVENSITY | KARLI

Funded by the European Union
NextGenerationEU

Supported by:
Federal Ministry for Economic Affairs and Climate Action
on the basis of a decision by the German Bundestag

Event driven cloud-based architecture for Data Centric AI development

Creators

Dr. Marc Großerüschkamp, Head of Software & Data Technologies

Mina Khosravifard, AI Expert

Created on

12.10.2023

INVENSITY | KARLI

Funded by the European Union
NextGenerationEU

Supported by:
Federal Ministry for Economic Affairs and Climate Action
on the basis of a decision by the German Bundestag

Abstract

A new research field in artificial intelligence (AI) is related to producing high-quality data and optimizing the speed of dataset creation from raw data. As industries move towards IoT and machine learning (ML), more and more raw data are collected. Examining this volume of data to generate datasets for ML models is time-consuming and costly and requires hardware with high computing power. Additionally, the artificial intelligence models used in the industry require constant updating in order to make use of the most recent available data. As a result, more than ever, there is a need for specialized tools to extract, transform and load (ETL) the raw data continuously and quickly into new datasets to retrain the machine learning models with these updated datasets. In this article, we show how an event-driven cloud architecture not only achieves the abovementioned requirements but also provides non-functional requirements such as scalability and efficiency.

Keywords: ETL, Event driven architecture, Cloud, EventBridge, DCAI, Small-to-big

1. Introduction

The advancements in deep learning in the last decade started a revolution that has attracted many researchers to work in this field. The early attention of researchers was on improving the accuracy of these models. This area of research is so-called model centric. While in the last two years, a new approach, the so called Data Centric Artificial Intelligence (DCAI), has been proposed to improve the accuracy of deep learning models by focusing on the quality of data and representative data that is less biased. Dataset creation is a costly requirement for such models, thus efficient data acquisition has a high priority. The DCAI approach has received much attention because it reduces the cost of producing the dataset, which is very noteworthy from an economic point of view.

KARLI is a current research project on the use of AI for adaptive responsive and level compliant interaction with the vehicle of the future (F. Diederichs et al., 2022) which is funded by BMWi (INVENSITY GmbH, 2022).

KARLI requires the acquisition of diverse data with a significant volume. For the training the AI models of this project a great amount of data shall be recorded. Several partners including research institutions and companies contribute to data acquisition, meaning that the data will not be collected in one physical place but instead distributed. Furthermore, the data will not be recorded all at once. Data will be continuously acquired in different periods. The result of data quality reviews will be used to improve the data collection in the next stage. We refer to

this approach of collecting data from "small to big" in the KARLI project. As soon as new raw data is available the quality of data shall be evaluated and the ETL mechanism shall automatically add the newly acquired data to the transformed dataset that can be used for training or retraining of the required ML models.

The continuous generation of a new datasets for ML models based on old and new data is another challenge with respect to data consistency. In KARLI we face the situation that ETL functionality is developed simultaneously with data acquisition. Therefore, an already available dataset might be transformed by an older version of the transform algorithm, while the newly acquired raw data will be transformed by the newest version of transform algorithm. Hence, the previously transformed data needs to be reprocessed in order to assure data consistency and allowing to easily use the continuously growing dataset for re-training the ML models. In order to maintain efficiency a high degree of automation and a well-defined orchestration of data evaluation and processing is mandatory.

Figure 1 illustrates the described scenarios.

Labeling of the dataset is not considered here as part of the dataset creation. As mentioned, we need to check the quality of the data, which will give us valuable information to improve the recording of the next stage after each data recording. This functionality can be integrated into the Extract part of the ETL system.

According to the several described needs the use of cloud services for storing, processing and distribution of data is the most promising approach in terms of efficient, consistent and scalable data management.

In summary the described use case of data acquisition for KARLI has three main requirements:

RQ I – SCALABILITY: It shall be possible to update the dataset, the transfer algorithm and the AI models continuously, and individually

RQ II – MONITORING: Actionable data monitoring shall be possible to allow a controlled data acquisition

RQ III – CONSISTANCY: Dataset consistency must be assured during continuous increas-ing the dataset (even though the transfer algorithm changes during the acquisition phase) Similar requirements have also been suggested for DCAI applications by (Polyzotis & Ma-tei, 2021)

INVENSITY
KARLI
Funded by the European Union
NextGenerationEU
Supported by:
Federal Ministry for Economic Affairs and Climate Action
on the basis of a decision by the German Bundestag

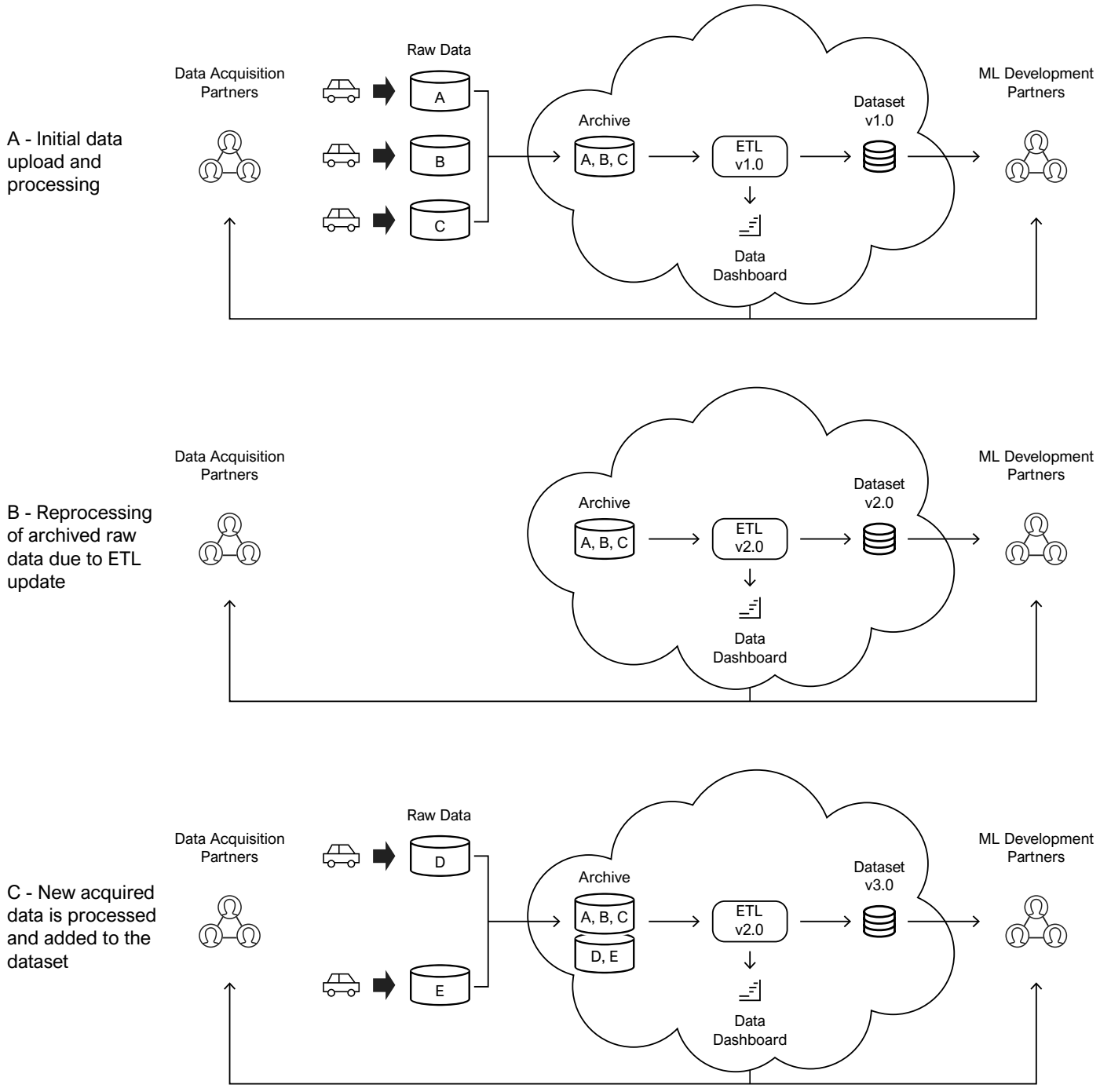

Figure 1: A - Initial data is uploaded from one or more project partners. The data is processed and then available for the ML development. B – In case the transformation algorithm changes, the dataset v1.0 is not consistent with the new ETL v2.0 and for this reason repro-cessing of the data is required. C – New acquired data is transformed and added to the Dataset. A data dashboard can be used to monitor the quality of data and ideally also data distribution to allow for a guided data acquisition throughout the project.

2. Related Work

As the industry moves towards the IoT and AI world, the amount of data produced and col-lected continuously grows. Cloud services provide services that speed up the development of data analysis software and eventually ease up AI development. Thorpe et al. used a scalable and serverless service called AWS Lambda for GNN training instead of purchasing and maintaining expensive GPUs. The thousands of lambda threads scale GNN training to billion edge graphs at a lower cost than infrastructure with GPUs (Thorpe, et al., 2021). Zhengchun et al. use Globus Flows (Chard et al., 2019), funcX (Chard, et al., 2020), and Globus file transfer (Foster, 2011) services to create a DCAI system for rapid re-training AI models within reasonable timescales. Interestingly they show that the turnaround time of their remote DCAI systems is still less than a nearby deployable GPU (Zhengchun, et al., 2021).

Pogiatzis et al. create an Event driven ETL pipeline using entirely serverless services using S3 (Amazon S3, 2022), SQS (Amazon SQS, 2022), lambda (AWS Lambda, 2022), and Dy-namoDB (Amazon Web Services, 2022) services for tabular data. This ETL architecture supports consistency, reliability, and acceptable performance for practical use in applications up to one event per second. They compare serverless implementation's cost versus server-based architecture (Pogiatzis & Samakovitis, 2021). These systems provide a platform for building applications without considering the server infrastructure (Serverless on AWS, 2022). Nevertheless, this simplification imposes a series of limitations that we will be discussed in this paper.

3. Architecture

This section provides an overview of the developed event-driven DCAI-ETL architecture (ED-DCAI-ETL). Figure 2 illustrates the overview of ED-DCAI-ETL and the services used within this architecture. All these services are from AWS. As explained in the introduction section, we will record the data only some at a time, but these data will be recorded gradually. Ac-cordingly, we need to define a cloud-based architecture for our ETL system. It can automati-cally consider new data transformation to a dataset and quality analysis as soon as it uploads into the cloud storage. We used S3 as a storage service that has scalability, data availability, security, and performance (Amazon S3, 2022).

Event-driven architecture is an architectural pattern that defines actions based on events. An event, in simpler terms, is a change in the status of the system, and uploading new data can be considered as an event. Thus, the

event-driven architecture is triggered by receiving an event, which will be used between decoupled microservices. One of the components of this architecture is the event producer. After the event producer produces the event, it is placed in the event router, which sends the event to the related event consumer based on the defined rules (Event-Driven Architecture, 2022). EventBridge is a serverless service that expedites building a scalable event-driven application in the AWS cloud.

It was mentioned in the introduction section that one of the challenges of implementing this system is that the ETL functionality will be developed and implemented simultaneously to the data acquisition, and we need to update the ETL logic at some point when more information is available. As soon as any change in the implementation of any functionality applies, old events shall be recreated. With the recreations of the old events, the old data that has already been entered into the S3 storage can be reanalyzed with newly updated functionali-ties. Fortunately, this need is anticipated in AWS Eventbridge, so the Event can be archived and replayed when needed later. To archive events, we need to define an event pattern. Events that match this pattern will be archived within a preservation period before they are discarded. We need to archive all the events related to uploading new files. Once an update appears in the functionality of ETL, it is necessary to replay events from the archive. As soon as the replay is required, it can be followed by indicating which archive to replay archived events. We used a separate AWS Lambda service to implement extract, transform and load services. AWS Lambda is a serverless service, meaning there is no need to provision or manage the server. The automatic provisioning enforces some constraints on runtime, memory, and payload size. The transformation operation can however require a longer runtime than the fifteen minute limit of AWS Lambda. Therefore, we used an EC2 unit. The EC2 service is not serverless, so we need to manage the server. Instead of applying changes to the data by AWS Lambda, transform Lambda takes over the task of controlling the EC2 computing server. Thus, the EC2 applies the data processes without having a runtime limit. As soon as the changes are applied, the EC2 sends an event to the Eventbridge bus. This event is then pushed into the transformation lambda, and as a result, the transformation lambda turns off this EC2 until another event arrives for data transformation.

INVENSITY KARLI
Funded by the European Union NextGenerationEU
Supported by: Federal Ministry for Economic Affairs and Climate Action on the basis of a decision by the German Bundestag

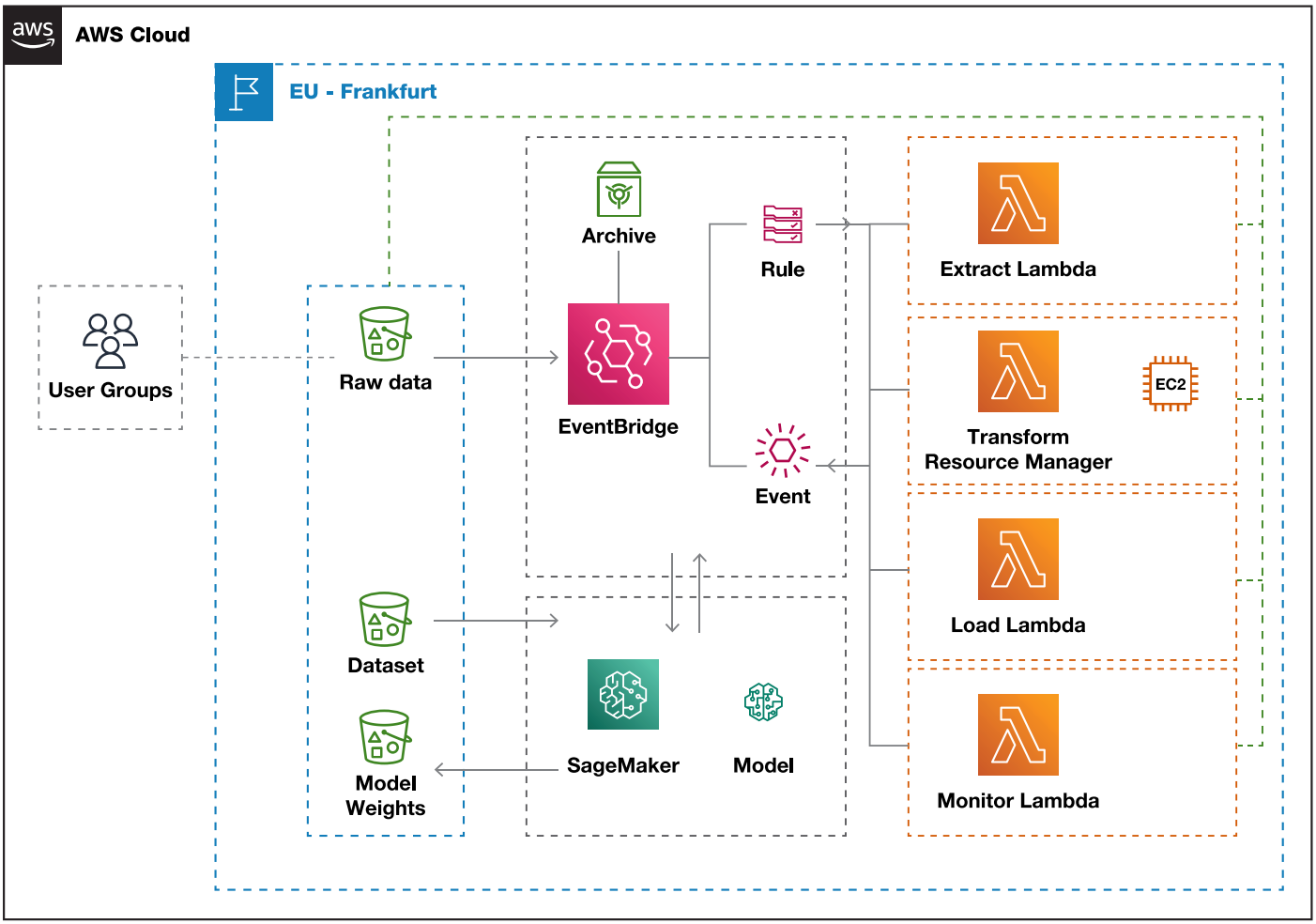

Figure 2: ED-DCAI-ETL architecture is an Event-driven architecture that events trigger the different services. Solid arrows show the events.

4. Results and Evaluation

The capabilities of the designed cloud architecture need to be assessed based on the described requirements

4.1 RQ I – Scalability:

We used event driven services, which means that when new data is entered into the Amazon S3 by one of the partners, it triggers the system to process the data. The processing cycle is as follows: Once any data upload into S3, the lambda extraction will be triggered by a notification event with name of event as create object. This computing resource reads the entered data and examines it for quality. It sends the result of its check as a custom event to the EventBridge bus. If quality of data is confirmed, this custom event has a value for the key, based on which this event causes the execution of the transform lambda. This lambda activates an EC2 computing resource and perform the necessary changes to convert the raw data into a dataset. Then, it stores the dataset in

the Amazon S3. After the completion of execution, EC2 sends a customized event with the processing key of end of process to the EventBridge bus. This event is sent to the transformation lambda and this lambda turns off the EC2 computing resource and finally sends a custom event with the content of the data set is available to the EventBridge bus. Lastly, this event forwarded the loading lambda, which select a part of the dataset for either (re)-training or for inferencing the AI model and store into the amazon S3.

Loading lambda eventually create a custom event and accordingly forward it either to Amazon SageMaker for training or testing lambda for inferencing purposes. Now, if the developer makes any change in the code, it is easy to get new data or a new model. When any user uploads new data, we archive an object creation event. As soon as the developer makes any changes to the code, it is enough to replay the archived events, and then the lambdas will be activated. According to the changes in the code, the dataset and the model will be automatically updated.

4.2 RQ II - Monitoring:

This architecture has a practical monitoring of the data in such a way that the control lambda receives all the events related to the data. For example, if low quality data is uploaded in Amazon S3, the extraction lambda checks the data and informs the problem in the form of an event, and this event is sent to the monitoring lambda, and this lambda informs the user in the form of a message in its output. This can approach can be extended for complete monitoring of data processing.

4.3 RQ III - Consistency:

Amazon S3 has data versioning feature to create different versions of a file. By enabling this feature, we ensure that every copy of the raw data can be retrieved, transformed into the desired format and stored in a separate storing location to be used for ML trainings. Consistency of the transformed data is achieved by the event replay mechanism described above. Consistency of ML model with respect to their performance and dataset version is ensure by the Amazon S3 versioning feature and a functionality developed for tracking the dataset version and training cycle.

5. Conclusion

In this study, we developed a cloud-based architecture for intermittent ETL for dataset (re)-creation. The core of this architecture accommodated with Amazon Eventbridge to support event driven architecture. We presented different cases to demonstrate the capability of this architecture for continuous dataset creation while the functionality of the ETL is still under development. We showed that this

architecture maximizes the automation in dataset creation for ML applications by informing the user about the quality of data and automatic data recreation after applying any changes or any new data entry. By combining serverless, full server and event-oriented services, we created processing units that are not limited in terms of memory and execution time unlike serverless services. This architecture can be used as blueprint for other machine learning projects and even can be extended for IoT projects to accept stream data as input data.

6. Funding

Supported by: Federal Ministry for Economic Affairs and Climate Action on the basis of a decision by the German Bundestag.

7. Abbreviations

AI - Artificial Intelligence
EC2 - Elastic Compute Cloud
AWS - Amazon Web Services
DCAI - Data Centric Artificial Intelligence
ETL - Extract-Transform-Load
GNN - Graph Neural Network
ML - Machine learning
S3 - Simple Storage Service
SQS - Simple Queue Service
IoT – Internet of Things

8. References

1. Amazon DynamoDB. (2022). Retrieved from AWS: https://aws.amazon.com/dynamodb/

2. Amazon S3. (2022). Retrieved from AWS: https://aws.amazon.com/s3/

3. Amazon S3. (2022). Retrieved from AWS: https://aws.amazon.com/s3/

4. Amazon SQS. (2022). Retrieved from AWS: https://aws.amazon.com/sqs/

5. AWS Lambda. (2022). Retrieved from AWS: https://aws.amazon.com/lambda/

6. Chard, R., Chard, K., & Foste, I. (2019). Globus Automate A Distributed Research Auto-mation Platform.

7. Chard, R., N. Babuji, Y., Zhuozhao, L., Skluzacek, T., Woodard , A., Blaiszik, B., . . . Chard, K. (2020). funcX: {A} Federated Function Serving Fabric for Science. CoRR.

8. Event-Driven Architecture. (2022). Retrieved from AWS: https://aws.amazon.com/event-driven-architecture/

9. F. Diederichs et al. (2022). Artificial Intelligence for Adaptive, Responsive, and Level-Compliant Interaction in the Vehicle of the Future (KARLI). 24th International Conference on Human-Computer Interaction, HCII 2022 (pp. 164-171). Virtual Event: Springer.

10. Foster, Ian. (2011). IEEE Internet Computing. Globus Online: Accelerating and Democ-ratizing Science through Cloud-Based Services, 70-73.

11. INVENSITY GmbH. (2022). Künstliche Intelligenz für eine adaptive, responsive und levelkonforme Interaktion mit dem Fahrzeug der Zukunft. Retrieved from https://karli-pro-jekt.de/

12. Pogiatzis, A., & Samakovitis, G. (2021). An Event-Driven Serverless ETL Pipeline on AWS. Applied Sciences.

13. Polyzotis, N., & Matei, Z. (2021). What can Data-Centric AI Learn from Data and ML Engineering? arXiv preprint arXiv:2112.06439.

14. Serverless on AWS. (2022). Retrieved from AWS: https://aws.amazon.com/serverless/

15. Thorpe, J., Yifan , Q., Eyolfson, J., Shen, T., Guanzhou, H., Zhihao , J., . . . Guoqing, X. H. (2021). Dorylus: Affordable, Scalable, and Accurate GNN Training with Distributed CPU Servers and Serverless Threads. 15th {USENIX} Symposium on Operating Systems Design, (pp. 495–514.).

16. Zhengchun, L., Ahsan, A., Kenesei, P., Miceli, A., Sharma, H., Schwarz, N., . . . Hong , C. (2021). Bridge Data Center AI Systems with Edge Computing for Actionable Information Retrieval. In 2021 3rd Annual Workshop on Extreme-scale Experiment-in-the-Loop Computing (XLOOP) (pp. 15-23).