Towards Situational Awareness with Multimodal Streaming Data Fusion: Serverless Computing Approach

Alina Nesen
Computer Science
Purdue University
West Lafayette, IN, USA
anesen@purdue.edu

Bharat Bhargava
Computer Science
Purdue University
West Lafayette, IN, USA
bbshail@purdue.edu

ABSTRACT

The availability of large quantities of data has given an impulse for methods and techniques to extract unseen useful knowledge and process it in a fast and scalable manner. In order to extract the most complete possible knowledge from the continuous data stream, it is necessary to use the heterogeneous data sources and process information from multiple modalities. The systems that utilize multimodal data must take advantage of the up-to-date approaches for data storage, usage, cleaning and storage. Neural networks and machine learning approaches are widely used for data-heavy software where pattern extractions and predictions need to be conducted while serverless computing frameworks are being increasingly used for machine learning solutions to optimize cost and speed of such systems. This work presents a framework for processing data from multimodal sources where the task of feature and pattern extraction is performed on a serverless computing platform. The use cases for public safety solutions to increase situational awareness are described and compared with other implementation approaches.

KEYWORDS

Serverless computing, function-as-a-service, multimodal machine learning

ACM Reference format:

Alina Nesen and Bharat Bhargava. 2021. Towards Situational Awareness with Multimodal Streaming Data Fusion: Serverless Computing Approach. In *Big Data in Emergent Distributed Environments (BiDEDE'21), June 20, 2021, Virtual Event, China.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3460866.3461769

1 INTRODUCTION

Big data and its analytics along with artificial intelligence in distributed environments are the crucial components of modern business, healthcare, financial, security and entertainment services. The abundancy of the incoming streaming data in different modalities be it text from social media or newsfeeds, video from surveillance cameras or numerical data from sensors requires fast, scalable and feasible solutions in order to make use of the knowledge and patterns in it [1]. From the design and implementation point of view, client-server architectures have proved to be efficient for deploying the neural networks for inferencing. However, server-based architecture requires a minimum commitment of resources at any given time since there needs to be some server running regardless of whether the system is being used or not; thus, there is a higher cost involved with such systems. Recently, serverless architectures for deploying and training neural network solutions have been proposed and shown efficient [2,3]. These services allow deployment of the software in the form of functions, hence the name function-as-a-service. The functions are executed in the infrastructure of the serverless computing environment in response to specific events such as new files being uploaded to a cloud data store, messages arriving in queue systems or direct HTTP calls. While the existing architectures that are described in the literature were designed separately for natural language processing applications and video applications, in this work we extend the applications multimodal data processing altogether, which includes video, text and sensor data with the far-reaching goals of designing an engine for a complete multimodal data framework that can be used to increase situational awareness (SA). This is the first attempt to implement a SA system that processes heterogeneous data with multiple machine learning models and takes advantage of the serverless architecture. The proposed framework has shown to have a number of advantages over the client-server approached that was developed in other works.

Situational awareness and systems that provide situational knowledge on demand play a crucial role in effectiveness and efficiency of business operations across variety of industries, from military and government to public safety and transportation. Being able to predict the influence of the events and actions that are happening on the future affairs is essential for on-the-spot decision-making. Typically, the systems that provide situational awareness process one or more feed of data of the same modality: there can be dozens of video cameras in one system or news feeds in another system but in a complete scenario, it is important to

ACM ISBN 978-1-4503-8465-0/21/06.
https://doi.org/10.1145/3460866.3461769

involve as many data modalities from the surroundings as possible in order to develop a holistic view of events. In [4] the authors describe the framework for situational knowledge on demand that includes video and text data combined that is processed with feature extraction modules and stored on a RDBMS server. The heterogeneous incoming data is processed and the distinct pieces of information are extracted from each modality in order to be combined together to form an event and complement to the situational awareness of the expert. For successful discovery of missing puzzles of the full picture in the situational knowledge on demand systems, it is necessary to continuously process incoming streaming data and update feature extraction models based on the detected objects and entities. Machine learning models for feature extraction need to be adjusted with the new data to avoid data drift, to understand the categories, detect novelties, calibrate their performance and retrain if needed. Implementing the multiple modules of the framework infrastructure is the most challenging part required for the successful system operation. Maintaining the system at all times involves human and monetary resources. Hence, taking advantage of the modern serverless solutions seems to be the optimal path when designing and engineering such systems. With this rationale, we aim to develop a prototype of a serverless framework for situational knowledge on demand which is rich in video, numerical and sensor data processing. Since serverless computing allows to perform cloud-based on-demand execution of feature extractions, we redesign the video feature extraction module presented in [4] to take advantage of the serverless platform. It has been noted that many organizations report significant cost savings by switching from traditional hosting operations to serverless solutions [23]. The new proposed design of the framework can be extended for text, structured and sensor data following the same procedures.

West Lafayette Police Department Use Case. Due to the nature of the forensics analysis, every police investigation includes large amounts of multimodal data. When searching for a person of interest, police detectives get information from the various surveillance cameras installed on the streets and nearby businesses as well as obtain text descriptions that come from incident reports or social networks. To combine the multimodal data sources and relate the frames from the video to specific description in the incidents, the appointed detectives must watch hours of video. This takes a lot of person-hours and may delay the discovery of important information or lead to a situation when a related event is missed because of inability for humans to process hours of videos and thus leave a mission unaccomplished. Automated process of mission-relevant knowledge discovery must possess high capabilities of multimodal data processing along with timely and accurate feature and attribute detection. A system that takes a description of an event (such as verbal narrative reported by a citizen to a police officer or a structured SQL-like query) must process the multimodal data (in our example, the modalities include text and video) from available sources to discover anything related to this event. We provide implementation details for processing the video modality in a serverless module with a trained neural network for object detection and recognition as well as a guidance for implementation with similar approaches for other modalities.

The paper consists of the following sections: Section 2 talks about related work both in the multimodal data processing for situational awareness and serverless computing for machine learning applications which are the main methods for knowledge extraction from the heterogeneous data. Section 3 provides a description of the framework and a roadmap for a domain- and platform-agnostic implementation of the platform. Section 4 contains detailed description of the framework for multimodal data processing. Comparative analysis of the three different approaches (server-based, serverless and hybrid) to implementation with regard to cost and performance is presented. We finish with conclusions, future work directions and description of the open challenges that remain in the domain of the situational knowledge on demand and detection of the mission-relevant data from multimodal sources Section 5.

2 RELATED WORK

Detailed description of the implementing of the frameworks for multimodal data processing and video querying with microservices and RDBMS servers are provided in [4,5]. The implementation approach in [4] was based on microservices usage as opposed to a function-as-a-service approach, hence in [4] the services are deployed and maintained by different owners and not controlled by the workflow developer. While this solution is also inherently distributed and some aspects of application management are done by an external entity (server owner or in part cloud provider), the workflow itself is organized in a different manner. [5] presents a comprehensive solution of video querying for the police investigation scenario. Another approach to process the incoming video data stream for hidden anomalies which can be potentially expanded to multimodal streaming data is given in [6]. However, that work does focuses on the server-based approach. The serverless aspect of the proposed framework is partially described in various works devoted to deploying scientific workload on serverless infrastructures, such as [7,8,9]. The authors provide review of the main providers of functions-as-a-service with their features, pricing and deployment limitations. In [10] the suitability of serverless computing to run deep learning inferencing tasks is evaluated. The response time, prediction time and the cost of executing the lambda function are evaluated and reported. Cost-effective approach of serverless computing may seem like an appealing reason for deploying machine learning based frameworks, hence the [11] proposes a method for the cost prediction, which consists of such factors as the response time of the function and the memory allocated to the function, of serverless workflows consisting of input-parameter sensitive function models and a monte-carlo simulation of an abstract workflow model. These cost predictions allow workflow designers to evaluate and compare workflow alternatives as well as optimize existing workflows. A pay-per-request deployment of convolutional neural network models for natural language processing using serverless services is also discussed in [26].

Along with that, [27] and [28] present video processing serverless solutions. Processing capabilities of computer vision solutions are benchmarked in [18]. An asynchronous distributed machine learning framework with a scheduler based on deep reinforcement learning is proposed in [17]. In [23] the companies migrating from traditional hosting to serverless solutions are studied. It is reported that cost savings reach as high as 66% and 95% after switching to serverless. In [12, 19, 27] serverless model training and the full workflow for building a serverless machine learning framework is evaluated from multiple perspectives. A general overview of modern challenges and opportunities in serverless computing is given in [13, 21, 29]. In [30] authors present a query execution engine built on cloud function services. Issues with distributed computing, open source and custom hardware are addressed in [14]. The performance issues due to the cold start of a container environment are discussed in [15] with a solution to pre-craft such resources and then to dynamically reassociate them with baseline containers. Another serverless platform to build and operate edge AI operations in edge cloud systems in presented in [16] and [21].

3 MULTIMODAL DATA PROCESSING FOR SITUATIONAL AWARENESS

The main parts for the system that involves massive data processing, analytics and fusion are the modules for data collection and cleaning, feature extraction with the help of pre-trained neural networks models or other methods, and extracting of the pieces of information of interest, be it video frames, text documents or any other events. Data fusion can be done at different levels of the pipeline: observation level, feature level or decision level. The central idea for optimal data fusion is identifying the criteria upon which the extracted entities should be combined into a joined object, entity or event of interest. These criteria can vary for various domains and applications and can be decided upon with the help of an expert or a human-in-the-loop. For example, for police investigation and forensics science scenarios, the similarity between the semantical content of the different modalities streams and the proximity of the temporal and spatial metadata in the extracted events and entities can serve as such criterion. Thus, the tweets that were issued at 2 am in the city of West Lafayette will be connected with the video frames collected in the cameras at that location around that time provided they are related to the same or similar topics.

Data Quality and Data Cleaning. It is estimated that the majority of time needed for building a good machine learning model is devoted to data preparation and data cleaning. The cleaning approaches should be carefully selected and tested to confirm that they are providing sufficiently cleaned data. In the simplest implementation, data cleaning can consist of procedures that remove invalid data, records with missing cells or frames with undetectable objects. A more advanced solution may contain methods and techniques for missing data completion and imputation.

Deep learning models for knowledge and feature extraction. The serverless situational knowledge on demand framework consists of the object storage, metadata catalogs, and distributed data processing services. The interface for the streaming data expects the video frame or a document text along with the meta data, which includes location and timestamp for static surveillance cameras or a subset of these data if the complete information is not obtainable. The output of the feature extraction submodule for every activated model is the list of the detected objects along with their attributes belonging to the frame or the similar sort of list for the document processing. In the wider sense, the knowledge extraction engine is fusing the extracted entities and attributes into the events and once an event is formed and completed, the classification output may be a binary label whether the event is of interest to the mission expert. Thus, the services will process HTTP requests containing raw input and the metadata and return back the extracted objects and their attributes. For instance, the trained model for video feature extraction (developed based on the YOLO neural network architecture) is placed in the same region as the code that reads and processes the frames in order to decrease latency access to the data storage. The data catalog in the serverless environment can store and organize together a collection of tables. In addition to data schemas, statistics about the tables can also be inferred from the stored data, such as the number of objects stored, number of rows in the tables, average amount of space occupied, and so on.

Proposed serverless architecture of the framework for the video and text processing with cross-analysis in the data fusion module has the following benefits:

1. Autoscaling that allows smooth running of the services to handle the spikes in the load regardless the number of the incoming requests. The appropriate capacity (memory, storage, bandwidth) is not the responsibility of the operations manager anymore.
2. As a consequence, the model can be served from a variety of front-ends such as web applications, mobile applications, and so on. This is especially notable for rapid mission-relevant information extraction in the framework for multiple users.
3. The storage service is deployed in the same infrastructure as functions, so data transfers become faster, arguably making the FaaS architecture more suitable for data-intensive workflows.
4. At the same time, the serverless solution comes with secure isolation which means that multiple users of the platform can work with different machine learning modules and code at the same time without interfering each other's efficiency.
5. Delegating the administration of the framework and infrastructure management together with monitoring and logging to the serverless computing establishment and allowing to focus on the logical part of the knowledge extraction engine.
6. Reusability and ease of extension for other modalities: this approach provides improved modularity as similar pipeline will function for the text or sensor data after necessary modification in the logics.
7. Over-provisioning of resources is avoided by fine spatial granularity (running functions with as few as 128MB of memory)

and fine temporal granularity (allocating just a few minutes of resource usage).

8. Cost savings for the pay-per-transaction plans and resource transparency. Users are billed only when the code is actually executed on the platform rather than resource allocation. In addition, functions as a service usually have a fine-grained and flexible pricing model.

9. Ease of monitoring and managing the lifecycle of the trained machine learning models and systems once they are in production.

10. Serverless computing promotes green computing by the on-demand execution of functions and releasing the resources after the execution and by billing per execution time, which incentivizes the programmers to improve the source code for higher efficiency.

4 SERVERLESS FRAMEWORK FOR SITUATIONAL AWARENESS

Overall architecture description. The mission-relevant situational knowledge on demand engine was designed to solve the use case of the West Lafayette Police Department to enable fast and reliable investigation and search for a given object with the pre-defined attributes from multimodal data.

Model training. The classes for multiclass classification were selected based on the most common mission needs in the city scenario: person, car, bus, bicycle, backpack, bag, and the related attributes, such as color, gender, apparel type; altogether we chose 30 classes of interest. The next step is to train a neural network to detect the specified class(-es) from multimodal sources, however, since not all the classes may be needed at the same time during a particular investigation, it is not efficient to apply a single large model that is trained to detect all possible classes from a video frame or document. For example, a certain user might only be interested in cars of specific color that were reported by citizens in incident reports or captured by the surveillance cameras in the city. Instead of utilizing the heavier model that detects all possible classes, which may introduce concerns regarding the model size limitations for the serverless engine, we train a set of smaller models for each data modality, and at the time of inference activate only the ones that are needed. In our example, that would be the model that detects cars and colors. For the initial implementation, we introduce 4 models: two for video object detection and two for text topic understanding. The user may activate the needed models that contain the classes of interest. This approach is cost effective with regards to both model training and model inference. The lego-like structure of the combined set of neural networks provides flexibility and room for extending the potential classes to be detected. For example, if the system is going to be deployed in the rural setting, only a smaller model that detects local unique classes has be designed and trained. This approach also allows to take advantage of the large selection of pretrained networks, as they may be added to the framework with minimum development time.

The data fusion module notifies the user about the frames or documents that contain both classes of interest and form a full event. Additional criteria that fuse the frames and documents into one event based on their proximity in location or time can be added (Fig.1).

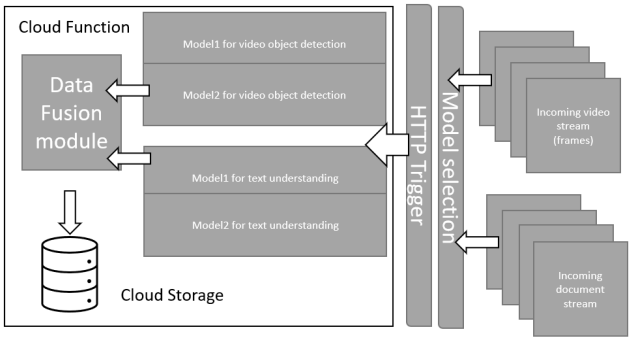

Figure 1. Serverless architecture for inferring the needed classes from multiple models. The needed events are eventually stored in the Cloud Storage.

A serverless function classifies the frames by performing a forward pass through the model. The proposed solution is platform-agnostic and can be installed using AWS Lambda, Google Cloud Functions, IBM OpenWhisk, MS Azure Functions or open source projects that provide infrastructure for event-driven stateless functions as microservices. The technical limitations may vary from platform to platform. For example, AWS Lambda limits the size of the deployment a compressed package to 50 MB however this limit can be overcome by utilizing the lambda layers feature. Thus, the libraries do not have to be included into the deployment package to keep its size smaller but can still be used in a lambda function. Another option to meet the deployment model size requirement is to reduce the number of layers in the machine learning model: one may replace fully connected layers with global average pooling layers or represent the weights for the fully connected layers as 16 bit floating point numbers which may decrease the size of the model more than twice.

AWS Pipeline for the incoming data stream. The video is stored in the Simple Storage Service (S3) bucket in the form of frames. Data in the buckets can be read, deleted, written from anywhere at any time. The user pre-defines the classes that need to be detected. This action triggers the inference Lambda function of the corresponding models. The same pipeline is applied to the text modality. Once the inference results are obtained, they are fused into one event of interest and stored in the persistence database store where it is indexed. The user notification is triggered about successful detection of the event in multimodal data sources.

Real-time predictions vs offline predictions. The scenarios for which the proposed framework was designed are currently targeted at offline predictions, when the historical video data is analyzed for pattern matching with specified query and data from other modalities. Real-time predictions however rouse higher situational awareness. For real-time predictions we implement the synchronous approach when the prediction and the response are performed in sequence between the caller and the ML model service. For a multi-user real-time situational awareness system, the optimal design involves asynchronous push/poll models where

predictions based on live streaming data are delivered to the user independently of the request for prediction, and the generated prediction is pushed as a notification. In the poll stage, however, the model generates predictions and writes them in a low read-latency storage which the caller periodically polls for predictions.

Performance limitations. Latency time and prediction time for the incoming data varies depending on the need to make predictions in real time or offline. In online predictions the current context and the historical information are being used to produce a result. The tradeoff between the predictive performance of the model and its prediction latency implies lower accuracy for real-time streaming systems. For real-time object detection YOLO v3 CNN [27] is preferred over RCNN for the speed vs. accuracy trade off.

Time analysis for AWS pipeline components. To determine which model has the best the performance in the framework, we repeat the experiments with three versions of YOLO neural network architecture for benchmarking. The general tendency is that models with larger inference time achieve better accuracy. However, since all the models were based on the same YOLO architecture, their accuracy and robustness differ very little. We also trained a custom neural network on the video dataset specifically collected from the West Lafayette Police Department. Figure 2 displays the average time taken to process a video frame in the testing stream of 1 hour of video when it goes through the framework pipeline for different configurations of object detection models. Each bar corresponds to a different type of object detection model with YOLO v3 being the heaviest and YOLO v4 the lightest in size.

Figure 2. Time needed to process a video frame at each submodule of the framework (in seconds). S3, SQS, Inference and Storage represent what proportion of time a video frame spends at each step in the pipeline.

As can be seen from the figure 2, the configurations of YOLO do not have a major influence on the result though the custom trained model performs the best. Each request writes the result of the inference with the list of classes names, corresponding bounding boxes and scores to the storage in json format. The last block represents the amount of time needed for a PUT request once the user has been notified of the detection of an interesting frame.

Figure 3 shows the ratio of response time of each deployed model to the local environment. The execution time of the local environment is set to one. Models for video inference are generally showing slower performance.

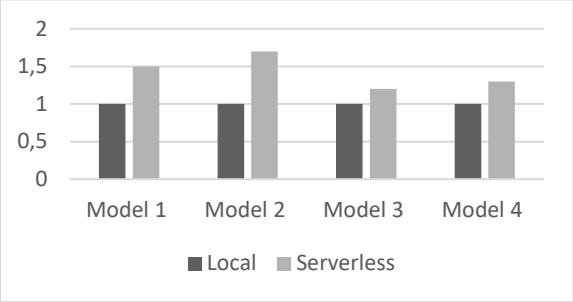

Figure 3. Response time ratio of the trained models. Local response time is normalized and the relative serverless time is displayed for each model. Model 1 and 2 are for object detection and Model 3 and 4 for text topic extraction.

Cost analysis. While for non-serverless approaches the cost is calculated for the entire time of services being in production, for the serverless implementation, such as AWS Lambda, the charge is only for the amount of time the functions are up and running. This makes serverless platforms extremely appealing for scenarios where the framework expected workload is uneven and the model sizes are relatively small. In the proposed framework, we distributed the attributes of interest into several models each detecting specific classes to make sure the size of each one does not exceed the quota.

5 CONCLUSIONS AND FUTURE WORK

In this work we have proposed a framework for the situational awareness system for heterogeneous data with serverless computing. While our solution is an early prototype, we find the presented approach very promising in terms of cost-efficiency and ease of development and deployment. As a general conclusion, the serverless approach can be beneficial in cases when the workload is uneven and unpredictable because it outsources the scalability issues to the third party while increase in the cost remains stable. However, in the absence of variations in the system's workload it might be optimal to use a proprietary server or a server in the cloud, such as EC2 of AWS. A combination of two approaches may also be considered with automatic switch between the two based on the workload monitor sensor. As a future work direction, we are extending the framework with a serverless recommendation system for the user who is operating with the framework that automatically learns from user's queries and pushes to him relevant information when it is streamed.

6 ACKNOWLEDGEMENTS

This work has been partially supported by the Northrop Grumman Corporation. We thank Dr. Michael Stonebraker of MIT for discussions and help over the course of this project. Contributions by members of REALM-SKOD group from MIT and Purdue have been useful in making progress. We thank Sgt. Troy Greene of West Lafayette Police Department for collaboration.

7 REFERENCES

[1] Groppe, Sven. "Emergent models, frameworks, and hardware technologies for Big data analytics." The Journal of Supercomputing 76.3 (2020): 1800-1827.

[2] Matt Crane and Jimmy Lin. 2017. An exploration of serverless architectures for information retrieval. In Proceedings of the 3rd ACM International Conference on the Theory of Information Retrieval (ICTIR 2017), pages 241–244, Amsterdam, The Netherlands.

[3] Feng, Lang, et al. "Exploring serverless computing for neural network training." 2018 IEEE 11th international conference on cloud computing (CLOUD). IEEE, 2018.

[4] Palacios, Servio, et al. "WIP-SKOD: A Framework for Situational Knowledge on Demand." Heterogeneous Data Management, Polystores, and Analytics for Healthcare. Springer, Cham, 2019. 154-166.

[5] Stonebraker, Michael, et al. "Surveillance Video Querying." (2019).

[6] Nesen, Alina, and Bharat Bhargava. "Knowledge Graphs for Semantic-Aware Anomaly Detection in Video." 2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). IEEE, 2020.

[7] E. Jonas, S. Venkataraman, I. Stoica, and B. Recht, "Occupy the cloud: Distributed computing for the 99%," Computing Research Repository, 2017

[8] M. Malawski, K. Figiela, A. Gajek, and A. Zima, "Benchmarking heterogeneous cloud functions," in Euro-Par 2017: Parallel Processing Workshops (D. B. Heras and L. Bouge, eds.), pp. 415–426, 2018. ´

[9] M. Malawski, "Towards serverless execution of scientific workflows - hyperflow case study," in WORKS@SC, November 2016.

[10] Ishakian, Vatche, Vinod Muthusamy, and Aleksander Slominski. "Serving deep learning models in a serverless platform." 2018 IEEE International Conference on Cloud Engineering (IC2E). IEEE, 2018.

[11] Eismann, Simon, et al. "Predicting the Costs of Serverless Workflows." Proceedings of the ACM/SPEC International Conference on Performance Engineering. 2020.

[12] Carreira, Joao, et al. "A case for serverless machine learning." Workshop on Systems for ML and Open Source Software at NeurIPS. Vol. 2018. 2018.

[13] Shafiei, Hossein, Ahmad Khonsari, and Payam Mousavi. "Serverless computing: A survey of opportunities, challenges and applications." arXiv preprint arXiv:1911.01296 (2019).

[14] Hellerstein, Joseph M., et al. "Serverless computing: One step forward, two steps back." arXiv preprint arXiv:1812.03651 (2018).

[15] Mohan, Anup, et al. "Agile cold starts for scalable serverless." 11th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 19). 2019.

[16] Rausch, Thomas, et al. "Towards a serverless platform for edge AI." 2nd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 19). 2019.

[17] Wang, Hao, Di Niu, and Baochun Li. "Distributed machine learning with a serverless architecture." IEEE INFOCOM 2019-IEEE Conference on Computer Communications. IEEE, 2019.

[18] Elordi, Unai, et al. "Benchmarking Deep Neural Network Inference Performance on Serverless Environments With MLPerf." IEEE Software 38.1 (2020): 81-87.

[19] Carreira, Joao, et al. "Cirrus: A serverless framework for end-to-end ml workflows." Proceedings of the ACM Symposium on Cloud Computing. 2019.

[20] Hall, Adam, and Umakishore Ramachandran. "An execution model for serverless functions at the edge." Proceedings of the International Conference on Internet of Things Design and Implementation. 2019.

[21] Eismann, Simon, et al. "A review of serverless use cases and their characteristics." arXiv preprint arXiv:2008.11110 (2020).

[22] Kaplunovich, Alex, and Yelena Yesha. "Refactoring of Neural Network Models for Hyperparameter Optimization in Serverless Cloud." Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops. 2020.

[23] Gojko Adzic and Robert Chatley. 2017. Serverless computing: economic and architectural impact. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. ACM, 884–889.

[24] Z. Tu, M. Li, and J. Lin, "Pay-per-request deployment of neural network models using serverless architectures," in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations, 2018, pp. 6–10

[25] L. Ao, L. Izhikevich, G. M. Voelker, and G. Porter, "Sprocket: A serverless video processing framework," in Proceedings of the ACM Symposium on Cloud Computing. ACM, 2018, pp. 263–274.

[26] M. Zhang, Y. Zhu, C. Zhang, and J. Liu, "Video processing with serverless computing: a measurement study," in Proceedings of the 29th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video. ACM, 2019, pp. 61–66.

[27] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[28] Jiang, Jiawei, et al. "Towards Demystifying Serverless Machine Learning Training." 2021.

[29] Müller, Ingo, Renato Marroquín, and Gustavo Alonso. "Lambada: Interactive Data Analytics on Cold Data Using Serverless Cloud Infrastructure." Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.

[30] Perron, Matthew, et al. "Starling: A scalable query engine on cloud functions." Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.