Leveraging Multi-Modal Data for Efficient Edge Inference Serving

Joel Wolfrath, Anirudh Achanta, and Abhishek Chandra
Department of Computer Science and Engineering
University of Minnesota, Minneapolis, USA
Email: {wolfr046, achan009, chandra}@umn.edu

***Abstract*—Real-time analytics over data streams is often performed on edge devices, which offer privacy guarantees and lower-latency responses compared to centralized processing in the cloud. Data streams originating from sensors, mobile phones, or IoT devices are diverse and span multiple modalities, including RGB videos from cameras, time series data from wearable sensors, and audio signals. Previous research has focused on optimizing the individual analytical tasks associated with each stream, with a special emphasis on deep learning, which is computationally intensive and may be used to analyze video streams, among other things. While advances in deep learning have significantly improved inference accuracy (e.g. for computer vision tasks), state-of-the-art models are not well-suited for edge computing environments. Novel approaches are required to substantially reduce the computational burden, since edge systems are heterogeneous and typically have fewer GPU resources available for inference with deep learning models. We show that leveraging data from multiple modalities can complement or sometimes even replace resource-intensive inference, while maintaining or enhancing accuracy. We present DAISY: a Data-Aware Inference Serving sYstem which leverages multi-modal data to increase inference accuracy by dynamically selecting an appropriate model for each request. We thoroughly evaluate the proposed approach using state-of-the-art models and real-world data, which shows an increase in SLO attainment up to 60%, with a corresponding increase in inference accuracy of 5%.**

***Index Terms*—edge computing, inference serving, deep learning, multi-modal data, video analytics**

I. Introduction

Data is increasingly generated in a geo-distributed manner due to the rising prevalence of smart devices. Historically, data generated by these devices could be streamed over the wide-area network (WAN) to a cloud data center to be processed and persisted. Performing analytics in the cloud is appealing due to its large compute capacity and resource elasticity; however, it is often infeasible for modern applications to rely exclusively on the cloud due to local privacy restrictions and WAN scarcity. The edge computing paradigm addresses these constraints by executing analytical tasks on machines in close proximity to the data generating devices, thereby addressing the network and privacy constraints.

Edge workloads frequently process data from a diverse set of devices, including video cameras, accelerometers, Bluetooth beacons, microphones, motion sensors, WiFi access points and temperature or air quality sensors. These devices produce data streams that span *multiple data modalities*, and typically perform analytics targeted at each individual stream. Multi-modal data sources underpin many useful applications, including traffic monitoring [1], child and elder care [2], [3], aquatic activity monitoring [4], autonomous vehicles [5], smart cities [6], [7] and surveillance [8], [9], [10]. These applications often have strict service level objectives (SLOs) and require *low-latency response times*, since they have public safety implications. This is challenging for edge computing, since modern applications process large volumes of data and leverage expensive computation to produce high-accuracy inferences. Many video analytics systems utilize deep learning models, which increases the need for efficient processing at the edge, given the computational demand. One analysis found that object detection models such as YOLOv3 [11] can only run at 21.5 frames per second on a Jetson TX2 module [12]. Transformer models, segmentation models, and recurrent neural networks can also be deployed at the edge and consume significant computational resources. Large transformer-based language models like BERT can require hundreds of milliseconds to perform a single inference on mobile phones [13]. As model complexity continues to increase for new technologies such as large language models, new algorithms will be required to maximize efficiency, especially for edge deployments.

Existing work on efficient edge analytics largely focuses on systems-level optimizations or the optimization of specific analytical tasks. One mechanism for reducing edge computation is to offload a subset of the data to the cloud for processing. Network-aware systems have been developed to strike a balance between performing computation at the edge and in the cloud [14], [15], [16], [17]. However, these approaches are still dependent on constrained WAN links and may have privacy implications for the end users. Other existing work focuses on reducing the computational burden of a single modality, e.g. deep learning over video streams at the edge [18], [1]. In multi-modal settings, many different kinds of models may need to be deployed to the edge, which presents scaling problems as the number of devices continues to grow.

In this work, we show that resource intensive models can be executed less frequently by leveraging the multi-modal nature of data at the edge. Dynamically selecting the best available modality or combining data from multiple modalities has the potential to produce higher accuracy and lower latency inferences. We first examine real-world use cases and the challenges associated with performing analytics at the edge (section II). Next, we examine the characteristics

979-8-3503-9566-2/24/$31.00 ©2024 IEEE
DOI 10.1109/CCGrid59990.2024.00053

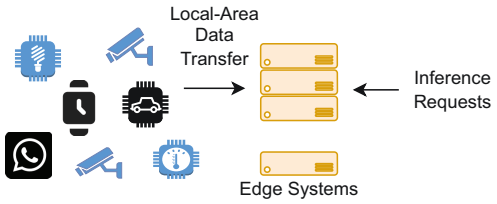


Fig. 1: Multiple devices stream data over a local network to edge nodes, which perform inference tasks.

of multi-modal models and compare and contrast the different modeling options available (section III). Finally, we propose and evaluate a data-aware approach (DAISY) for exploiting multi-modal data sources to dynamically select the best inference models based on the incoming data (sections IV and V).

Contributions. We make the following research contributions:

- We characterize and evaluate models that span multiple modalities and derive actionable insights.
- We show that selecting models based on *average* inference accuracy is suboptimal, since model accuracy varies substantially based on the data modality and target label.
- Using this information, we design a novel data-aware inference serving system (DAISY) which utilizes properties of the data to improve dynamic model selection while attaining high accuracy and respecting the target SLO.
- We thoroughly evaluate our proposed system using real-world multi-modal data and examine the sensitivity to various system parameters.

We show that incorporating multi-modal data and data-awareness into the model selection process can produce higher accuracy inferences while respecting the requested SLO.

II. Preliminaries

A. Motivating Applications

Patient Care. In the healthcare domain, edge analytics can aid patient monitoring and assistive technologies (in-home or in a healthcare facility). Video data can be utilized for activity recognition of elderly individuals to ensure their safety and well-being [2]. In the United States alone, over 30% of individuals over the age of 65 fall annually, which can be an emergency situation and contribute to permanent disability [19]. Patients may also have wearable sensors that capture data regarding their movements or general well-being [20]. By automatically detecting falls, abnormal movements or emergency situations, analytics systems can trigger timely alerts or interventions. Moreover, they can support rehabilitation programs by tracking and analyzing patient movements and gait, providing objective feedback. Available data modalities in the patient care domain may include RGB video frames [2], audio signals, and time series data generated by wearable sensors such as accelerometers [20], [21].

Smart City. Edge analytics plays a crucial role in smart city initiatives [22], enabling traffic management, waste management, public safety, and urban planning by analyzing data from various sensors and cameras. In the surveillance domain, video analytics plays a vital role in enhancing security and public safety [8]. By automatically identifying specific actions or abnormal behaviors, surveillance systems can alert security personnel to potential threats, thus enabling proactive response and crime prevention. Surveillance also encapsulates other aspects of monitoring to capture the dynamics of different application domains including retail, transportation, and service industries. Transportation hubs can benefit from the localization of passengers or nearby users to ensure they are operating effectively [23]. Multi-modal smart city applications also exist for individuals with Autism Spectrum Disorder (ASD) or those who are blind & visually impaired (BVI). For example, solutions have been developed that combine localization data from mobile applications and low-energy sensors to assist users with indoor and outdoor navigation [24].

Available data modalities in the smart city setting may include RGB video frames [8], audio signals, and localization data based on Bluetooth beacons [6], [24] or WiFi signals [3], [23]. Time series data is also prevalent and can be generated by devices such as motion sensors, smart phones [23], and temperature or air quality sensors [25].

B. Problem Characterization

There are many challenges associated with developing low-latency, multi-modal edge analytics systems, including:

- *Compute Intensive Workloads*: Video frames are often processed by deep neural networks, which are resource intensive [26]. Some systems also perform continuous learning, which increases the computational demand [27], [1]. Large language models may be deployed at the edge in the coming years, which have been shown to perform increasingly better as the number of parameters increases [28].
- *Data Volume*: As the number of smart devices grows, the total volume of generated data also necessarily increases. One estimate suggests there will be 75 billion connected devices by the year 2025 [29].
- *Heterogeneous Edge Resources*: Edge computing must contend with heterogeneous compute and network resources, which are often restrictive when compared to the resources and elasticity associated with cloud deployments [2].
- *Low Latency Responses*: Edge systems provide low-latency analytics, since they reduce dependencies on the WAN for communication. Furthermore, our motivating applications both involve user safety, which is critical to identify and remediate in real-time [30], [22].
- *WAN Scarcity*: Some edge systems leverage the cloud to perform inference or model retraining. These systems must contend with heterogeneous WAN links [14], which may constrain the amount of data that can be uploaded.

- *Privacy Constraints*: Applications that offload data to the cloud must consider how local privacy restrictions affect system operations. Video recordings used for analytics may contain personal and private user information, leading to serious privacy concerns on how this data is handled.

Prior works address some of these challenges; however, the presence of multi-modal data presents additional challenges and opportunities to increase the efficiency of edge systems. The possibilities are illustrated by considering the modeling options available to us.

Single-Modality Models. Existing inference serving systems provide access to single-modality models, i.e. models that take a single type of data as input. This includes models for image processing, time series data, graphs, etc. For example, suppose we are given an inference request to measure human occupancy (e.g. how many people are present in a given room). Existing systems rely on fixed protocols for generating an inference, which may include running video frames through an object detection pipeline (e.g. with a Yolo model).

Multi-Modal Models. Researchers have also designed models that operate on multiple data modalities simultaneously [31]. These models draw inspiration from the human brain and its ability to use multiple sensory inputs and contextual information to perform object recognition. For example, the MuMu [32] inference model can concatenate both time series data and RGB video pixels, which is then processed with a single deep neural network to produce a result. These techniques tend to be more computationally intensive, but have the potential to produce higher accuracy results.

Problem Statement. Our setting is a standalone edge system which processes a set of data streams $d_j \in \mathcal{D}$ which span multiple modalities. We assume we are given a set of models $m_i \in \mathcal{M}$, each of which operates on a subset of the available streams. The system provides *model-less* inference serving, where users issue inference requests to the system, but do not provide a model to execute. The system is required to select an appropriate model to serve each inference request. Each request has an associated target latency (which is our SLO). Our objective is to develop efficient model selection strategies, which maximize the inference accuracy while responding within the provided target latency. More formally, our model selection objective is given by:

$$\hat{m} = \underset{m_i \in \mathcal{M}}{\arg\max} \quad \mathbb{E}[Accuracy(m_i) \mid \mathcal{D}] \tag{1}$$

$$\text{s.t.} \quad \mathbb{E}[\ell(m_i)] \leq T \tag{2}$$

where $\ell(m_i)$ approximates the latency for executing model m_i, given the current system state and T is the target latency (SLO) for a given request. We focus on the human action recognition task, which captures aspects of both of our motivating applications (patient care and smart cities). We first conduct an exploratory analysis to obtain insights into multi-modal inference and motivate our proposed approach.

III. Characterizing Multi-Modal Inference

Inference serving for multi-modal applications presents several interesting trade-offs. For example, given the task of human action recognition, it is possible to classify human actions using a video stream; however, it is also possible to obtain high accuracy using exclusively accelerometer data generated by a smart watch [33]. Furthermore, it may be useful to combine both data modalities and perform inference using a *multi-modal* model, which has the potential to produce the highest accuracy inferences. Given the variety of options, we need to explore the trade-offs between these models to understand how they may be leveraged in an inference serving system.

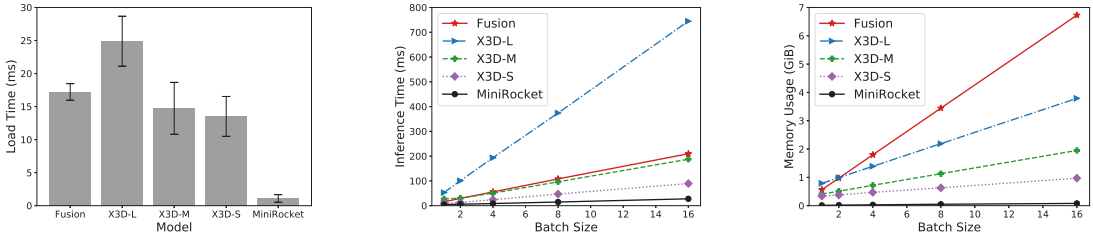
We briefly evaluate a variety of single-modality models for time series sensor data and RGB video data, along with one multi-modal fusion model. The task is to classify human action using one (or more) of these data modalities. We examine how these models compare in terms of resource usage and inference accuracy using the MMAct [33] dataset. The target class labels can be found in table I. We use X3D [34] for the video modality and MiniRocket [35] for the multivariate time series data generated by the sensors. The video-only models are the X3D-S, X3D-M, and X3D-L variants, which offer various compute/accuracy trade-offs. As a reference point, the X3D-M variant has 3.9M trainable parameters. All tunable hyperparameters were set to the same values in the original paper implementation. The MiniRocket model for time series data is a recently developed method that uses random convolutional kernels to transform the input time series data and trains a linear classifier on this transformed data. For a multi-modal fusion model, we adopt the architecture proposed by Choi, et al. [31], which performs feature extraction from each modality and fuses the results.

Resource Usage. Figure 2 shows the performance characteristics of the X3D family of video models, the MiniRocket time series model, and the fusion model. We observe that the MiniRocket time series model is substantially less resource intensive (for both compute and memory) compared to the X3D family of video models. For example, at batch size 8, MiniRocket is 7x faster for inference compared to X3D-M and requires 22x less GPU memory. Within the X3D family, we also observe noticeable differences in performance between the small and large models. For the fusion model, we observe that the load time and inference time is roughly comparable to the X3D-M model; however, the GPU memory usage can be much higher since it requires multiple data modalities to reside in GPU memory. This diversity in performance suggests that modeling choices in a multi-modal setting can greatly affect resource usage and throughput. Additional work is required to ensure these differences are considered during model selection.

Inference Accuracy. We now examine the differences in inference accuracy across models. We trained each of the following

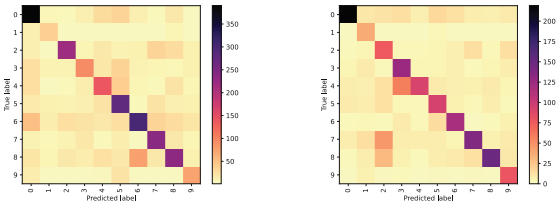
Labels (0-9)	carrying	kicking	talking on phone	entering	exiting	loitering	pulling	using phone	talking	throwing
Labels (10-19)	looking around	carrying (heavy)	carrying (light)	transferring object	checking time	setting down	closing door	sitting down	standing up	opening door
Labels (20-29)	picking up	running	pocket in	pocket out	crouching	falling	standing	jumping	using PC	waving
Labels (30-34)	sitting	pushing	pointing	drinking						

TABLE I: Target labels for MMAct Human Action Recognition Classification



(a) GPU Load Time (b) Model Inference Time (c) GPU Memory Usage

Fig. 2: Performance Characteristics for Single-Modality Models



(a) MiniRocket (b) X3D-M

Fig. 3: Confusion Matrices for MiniRocket and X3D-M

models on the MMAct [33] dataset. First, we consider the average F1 score that we obtained for each model on the human action recognition task (table II). We observe notice-

Model	Modality	F1 Score
MiniRocket [35]	Time-Series	0.61
X3D-S [34]	RGB Frames	0.64
X3D-M [34]	RGB Frames	0.65
X3D-L [34]	RGB Frames	0.68
Fusion [31]	Time-Series + RGB	0.72

TABLE II: Average F1 score across various models.

able differences in average F1 score across models. However, examining the average misses a key property: *model accuracy varies substantially based on target class label.* For example, there are target classes where the time series MiniRocket model is extremely accurate, and other classes where it performs poorly. Figure 3 shows partial confusion matrices for MiniRocket and X3D-M. We observe that X3D-M struggles to classify target label 4 ("exiting") while MiniRocket does not have this issue. Conversely, MiniRocket does a poor job with label 6 ("pulling"), while X3D-M is quite accurate. Table III highlights the target classes with the largest differences in F1 score between MiniRocket and X3D-M.

While it is possible to compare MiniRocket and X3D based purely on average accuracy, that misses the fact that *these models can be complimentary, depending on the data*. This motivates model selection strategies that leverage properties of the underlying data to maximize inference accuracy. Existing strategies make decisions based on average accuracy, which is inefficient in the multi-modal case. The performance of each model heavily depends on the target class. For these reasons, data sources need to be considered when making inference serving decisions. Selecting the right data source for the task is crucial for maximizing accuracy.

Action Label	Better Model	Absolute Difference in F1 Score
Kicking	X3D-M	0.37
Talking on Phone	X3D-M	0.23
Entering Room	MiniRocket	0.31
Loitering	MiniRocket	0.28

TABLE III: Labels with the largest difference in accuracy between the X3D-M video model and the time series model.

IV. DAISY: Data-Aware Multi-Modal Selection

We have shown that inference accuracy for each model depends heavily on the underlying data and target label. Given this characteristic, we now propose data-aware mechanisms for dynamically selecting an appropriate modality and inference model for each inference request.

A. High-Level Design

In our proposed system (figure 4), data is streamed to the edge, where recent data is kept in memory and stale data is either offloaded or persisted to disk. Our system has two main components for handling inference requests: a data-aware model selector (section IV-B) and a multi-modal SLO-Aware model selector (section IV-C). When a request arrives, our system first attempts to make model selection decisions based on the data. If the data-aware selector is unable to satisfy the request, then we defer to the multi-modal SLO-Aware approach to select and execute a model. As part of the model selection, the state of the system is incorporated (through our latency function ℓ) in order to estimate the expected

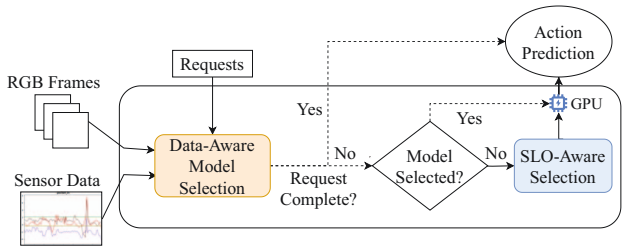


Fig. 4: Holistic view of the edge inference serving system.

latency. Selecting an implementation of ℓ is a complimentary research direction, which often involves offline profiling. Our system measures several standard sources of latency, including inference latency, GPU load time, GPU memory contention, and work queue length.

B. Data-Aware Model Selection

Our key insight is that *models that span multiple data modalities often provide complimentary benefits.* For example, although we observe cheaper time series models have lower average accuracy compared to video models, *the time series model excels at a certain proportion of the target classes.* To effectively leverage the cheaper models, we require a mechanism to determine in which cases the cheaper models suffice. Toward this end, we propose executing a lightweight model, which performs model selection based on the properties of the data. We consider two approaches: (1) specialized model training and (2) using an existing model with a threshold.

1) Specialized Model Training: We first consider training a specialized model exclusively for the task of model selection. This approach takes the available data as input, and performs a classification task, where each class represents a model which could be used for inference.

Analysis. For the specialized model approach, the additional delay incurred for model selection is the expected inference latency of the specialized model. More formally, for a given request R, the expected latency for the request is given by:

$$\mathbb{E}[\ell(R)] = \mathbb{E}[\ell(m_{sp})] + \mathbb{E}[\ell(\hat{m})] \tag{3}$$

where $\ell(m_{sp})$ is the inference latency for the specialized model and $\hat{m}$ is the model selected to service the request. Therefore, this scheduling mechanism always incurs a fixed overhead for model selection, which depends on the complexity of the specialized model.

2) Threshold (an existing) Model: Specialized models require each application to manually train one or more models offline for this very specialized use case and incur an overhead for each inference request. To avoid these costs, we consider another approach for dynamic model selection. We propose speculatively executing the lowest latency model available ($m_{low} \in \mathcal{M}$) and use that output directly *if the model confidence is sufficiently high*, i.e. if the class probability of the target label exceeds a specified threshold. For example, if we specify a threshold of $\tau = 0.5$, we can examine the class probabilities from m_{low} and if any of them exceed 0.5, we can just use the output from m_{low} directly, and avoid executing any other models. If none of the class probabilities exceed the threshold τ, then we defer to a multi-modal SLO-Aware approach (section IV-C).

Analysis. When applying a threshold to the lowest-latency model (m_{low}) for model selection, we obtain the following expression for the expected latency of request R:

$$\mathbb{E}[\ell(R)] = \mathbb{E}[\ell(m_{low})] + (1-\theta)\ \mathbb{E}[\ell(\hat{m})] \tag{4}$$

where θ is is the proportion of queries that can be answered accurately by m_{low} and $\hat{m}$ is the model selected by the SLO-Aware approach if the threshold was not met.

As previously discussed, the threshold model approach has the added benefit of being able to simply use the time series model output directly if the confidence is high enough. However, if the confidence is low, we require the execution of two models. This ensures higher accuracy, but raises the following question: *under what conditions does this strategy provide an expected request latency less than that of a standard SLO-Aware approach?* We are interested in cases where the latency is less than or equal to the expected inference latency for the best model choice $\hat{m}$. Using equation 4, we can derive the following requirement:

$$\mathbb{E}[\ell(m_{low})] + (1-\theta)\ \mathbb{E}[\ell(\hat{m})] \leq \mathbb{E}[\ell(\hat{m})] \tag{5}$$

$$\mathbb{E}[\ell(m_{low})] \leq \theta\ \mathbb{E}[\ell(\hat{m})] \tag{6}$$

This result suggests the threshold model must have at least one of the following properties:

- Its expected inference latency relative to the other available models must be low
- The proportion of queries that the threshold model can answer accurately (θ) must be high.

In our previous experiments, we observed that the inference latency for the MiniRocket model was substantially lower than all other models, which makes it a good candidate for a threshold model, given these criteria.

Selecting a Threshold. This approach requires a specified threshold (τ). Many existing applications (e.g. public safety) may already have requirements for model confidence, which can be used directly. If the application does not specify a hard requirement, users should select the threshold which maximizes inference accuracy, i.e. by solving:

$$\tau = \underset{t \in (0,1)}{\arg\max} \quad \mathbb{E}[Accuracy(m_{low},\ t)] \tag{7}$$

In practice, τ can be found using out-of-sample data and a grid search. If the highest accuracy selection of τ results in higher latencies than a baseline approach, this indicates that

the threshold model is not accurate enough to use for model selection (i.e. that θ is low in equation 4). In our experiments, we found that a threshold of $\tau = 0.5$ maximized accuracy with the MiniRocket model and it could answer a sufficiently large number of queries with comparable accuracy to the X3D models ($\theta = 0.42$). Furthermore, we observed that setting $\tau = 0.5$ yielded a threshold model with 87% accuracy[1].

In our system, we prefer the threshold model over the specialized model. In practice, this avoids requiring the user to train a specialized model for the system. Furthermore, threshold models have the ability to reuse the output for inference, potentially avoiding any additional model execution.

C. SLO-Aware Multi-Modal Selection

If the data-aware approach fails to produce an inference or assign a model, we default to an SLO-Aware approach for model selection. This approach proceeds in two steps:

1) When a request arrives, we filter the available models to find a subset which can satisfy the SLO. This may require the system to estimate execution latency for each model based on the number of outstanding requests and any potential resource contention.
2) Select the model which has the highest average accuracy within the filtered subset.

This approach incorporates scheduling decisions directly in the model selection process. Determining which models can meet the SLO necessarily requires knowledge of where the model will be executed and any delays incurred prior to processing. Since the edge is constrained, the number of possible executors is typically small, so a brute force approach suffices [15], [36].

To apply this strategy to the multi-modal setting, we require a mechanism for jointly selecting a data source in addition to a model. We propose performing filtering and selection across paired instances of data sources and models (figure 5). The candidates can be generated by computing the Cartesian product of the set of models and the (power) set of input modalities. The filtering step can then consider all generated pairs to see which ones meet the requested SLO. At first glance, this seems much less efficient, since the filtering step is now linear in $\mid \mathcal{D} \mid \times \mid \mathcal{M} \mid$ rather than $\mid \mathcal{M} \mid$. The key insight is that *the actual feasible set of data source/model pairs will be sparse, in the sense that many of the data/model combinations will be spurious.* In certain cases, the same model structure can be used across multiple modalities, but image data, for example, requires a specific set of models for inference (time series models are not applicable in any meaningful way). The number of model / data combinations that need to be considered is still approximately linear in the number of models; therefore, the time required to estimate inference latency remains unaffected. The full DAISY model selection algorithm is outlined in algorithm 1.

Algorithm 1: DAISY Threshold Model Selection

Input: Request R, Set of Models $\mathcal{M}$, m_{low}, Threshold τ, Input Data $\mathcal{D}$

```
// Data-Aware Model Selection
class_probs ← infer(m_low , D)
if max(class_probs) > τ then
    R.label ← label corresponding to max(class_probs)
    R.complete ← True
    return
end

// SLO-Aware Model Selection
candidates ← subset of M that attains R.target_latency
if candidates ≠ ∅ then
    m̂ ← argmax_{m_i ∈ M} E[Accuracy(m_i)]
    // Use mhat for inference
else
    // No time to execute a different model
    R.label ← label corresponding to max(class_probs)
end
```

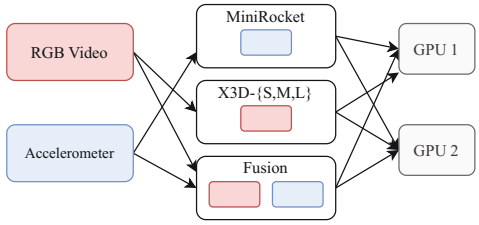


Fig. 5: Data sources and models can be combined and scheduled for inference as a single unit.

V. Evaluation

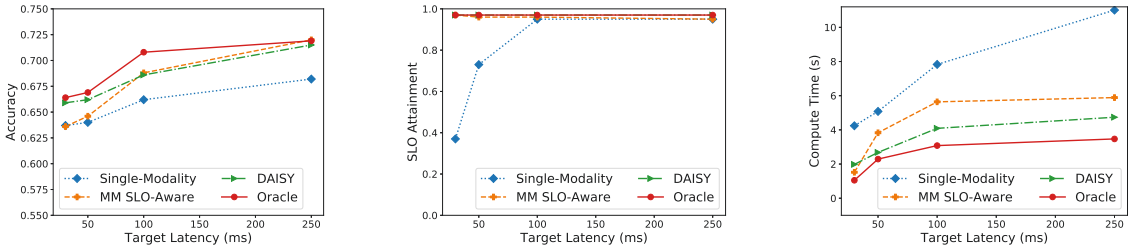
A. Methodology

Dataset and Queries For this evaluation, we use the MMAct dataset [33], which consists of video, acceleration, gyroscope and orientation information. The environment used is an indoor room, with various everyday objects placed in view. RGB video frames are the first modality, and are collected from four cameras around the room. Each participant also generates signals from an accelerometer on their hand, along with an accelerometer and a gyroscope in a smartwatch.The specific sampling rates and dimensions for each modality are shown in Table IV.

Mode	Sampling Rate	Dimensionality
Video	30 frames / sec	1920 x 1080, RGB
Accelerometer (x2)	100Hz	3-axis
Gyroscope	50Hz	3-axis
Orientation	50Hz	3-axis

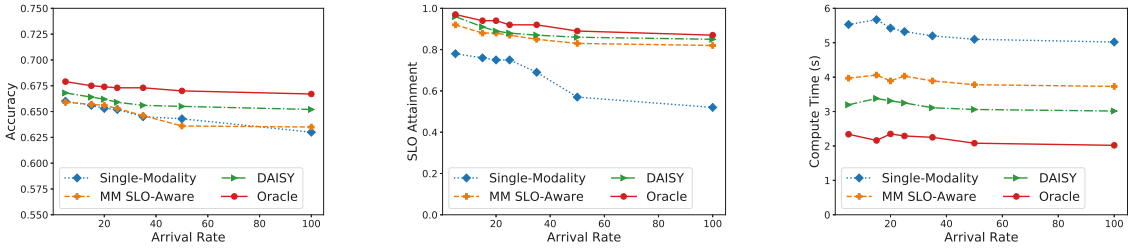
TABLE IV: Data Modes in MMAct

[1]The false positive and false negative rates were 0.09 and 0.02 respectively.



(a) Accuracy (b) SLO-Attainment (c) Compute Cost

Fig. 6: Comparison of model selection strategies across a variety of performance metrics.



(a) Accuracy (b) SLO Attainment (c) Compute Cost

Fig. 7: Comparison of model selection strategies across a variety of arrivals per second (following a Poisson process).

There are a total of 35 action classes that must be distinguished. The dataset is also divided into four scenes, with each scene having multiple sessions consisting of a participant performing various actions.

Testbed. Our experiments are conducted on a system with an Intel i5 CPU, 32 GiB of main memory, and an NVIDIA RTX 3060 graphics card, which has 12 GiB of memory. Video and time series data are read out of separate files that correspond to the same user in the same physical environment.

Baselines and Metrics. We compare the following model selection techniques:

- *Single-Modality (Video Only)*: Existing systems, which use only a single modality to serve inference requests (e.g. RGB video frames). These systems (e.g. LayerCake [15] and InFaaS [37]) filter DNN video models and choose the best available model which satisfies the SLO.
- *Multi-Modal SLO-Aware*: Our proposed modifications to the single-modality technique which can use multiple data modalities to satisfy a request and dynamically selects both a data source and a model. This includes using the Mini-Rocket time series model and fusion model to make inferences traditionally only offered by video models.
- *DAISY*: Our proposed system, which leverages data-awareness (with a threshold model) in addition to multi-modal SLO-Awareness when necessary.
- *Oracle*: An oracle which picks the optimal inference model with no overhead.

We evaluate each technique across multiple dimensions, including inference accuracy, SLO attainment, and resource usage. Unless otherwise specified, the default arrival rate is 10 requests per second, following a Poisson process. We examine other arrival rates in the sensitivity analysis.

B. Inference Accuracy

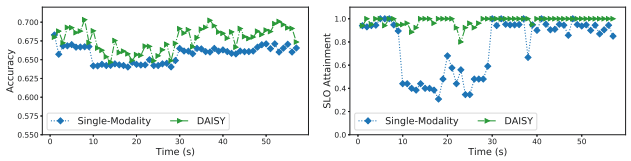
Figure 6a shows the average accuracy obtained by each method across a variety of target latencies. We observe that both of our multi-modal techniques obtain higher accuracy than the Single-Modality baseline, between 2-6%. This improvement is attributable to the data-awareness mechanism and the ability to leverage higher accuracy fusion models when the target latency is high. We also observe that both of our approaches are comparable to the Oracle when the target latency is sufficiently high. In this case, all of our multi-modal approaches can afford to execute the high-accuracy fusion model for almost every request.

C. SLO-Attainment

Our objective of maximizing accuracy was also constrained, since the user may specify a target latency (SLO) associated with each request. Figure 6b shows the percent SLO attainment for each method across a variety of arrival rates. We observe that the Single-Modality baseline misses the SLO with very high frequency for target latencies less than 100ms. The multi-modal approaches do not have this problem, since they have a very low-latency time series model, which can be used for inference when the target latencies are low.

D. Compute cost

We now consider how each model selection strategy affects the overall computational cost to service a set of requests.



(a) Inference accuracy (b) SLO attainment

Fig. 8: Dynamic Arrival Rates

This includes the CPU/GPU time required for model selection and the time to perform inference with any additional models. Figure 6c shows the resulting compute times associated with various target latencies. We observe that the mutli-modal approaches achieve higher accuracy and higher SLO attainment while using less compute resources. This is due to the time series models providing accurate inferences for a subset of requests with very low latency.

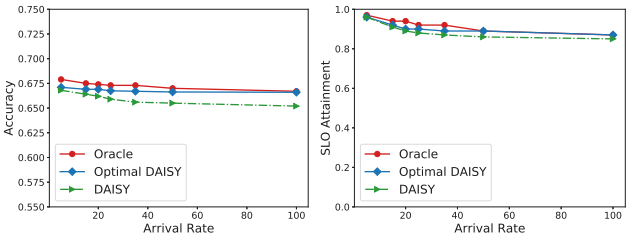
E. Sensitivity Analysis

Arrival Rates In practice, inference serving systems will need to support different workloads and query frequencies. We now examine how our system performs across different arrival rates, with a fixed SLO of 50 milliseconds. Figure 7a shows the average accuracy obtained by each method across a variety of arrival rates. We observe that both data agnostic methods achieve similar accuracy (with a slight improvement for the multi-modal approach). DAISY improves over both of these techniques, with an increase in accuracy between 1-5%. This improvement is attributable to the data-awareness mechanism, which attempts to exploit the accuracy differences across different models and target labels.

Figure 7b shows the percent SLO attainment for each method across a variety of arrival rates. We observe that both of our proposed methods improve substantially over the baseline Single-Modality approach, a 20% improvement for the multi-modal SLO-Aware approach and 29% for DAISY. Our approaches are able to leverage cheaper time series models when the arrival rate becomes high, which accounts for the difference in SLO-Attainment.

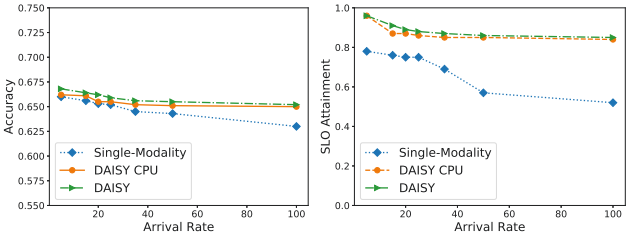
Figure 7c shows the resulting compute times associated with various arrival rates. We observe that DAISY uses substantially less compute resources compared to the data-agnostic approaches. This is due to its ability to use the threshold model output directly for inference. We also observe that for a given technique, compute times are largely consistent across arrival rates. This is expected, since for each experiment, the system processes a fixed number of requests.

We now examine how dynamically changing arrival rates affect accuracy and SLO attainment. We initially fix the arrival rate at 10 requests per second, then we introduce a spike in requests (100 requests per second) for a duration of 20 seconds before returning to the original rate. Figures 8a and 8b show the results for this experiment. For accuracy, we see that both the Single-Modality baseline and DAISY experience reduced accuracy during the request spike. This is



(a) Accuracy (b) SLO Attainment

Fig. 9: Optimal DAISY Selection



(a) Accuracy (b) SLO Attainment

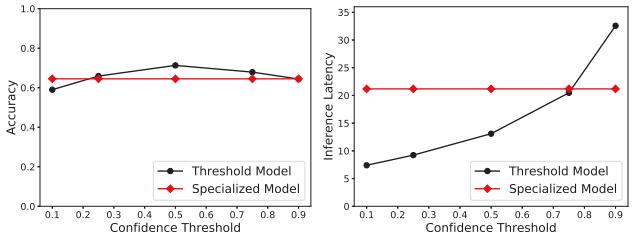
Fig. 10: CPU Threshold Model

expected, as both systems attempt to execute faster models to meet demand. For SLO attainment, we observe that the Single-Modality baseline is severely impacted, attaining between 40-60% of the SLOs. With a few minor exceptions, the DAISY approach maintains very high SLO attainment, even during the higher arrival period.

Threshold Model Performance. We now compare our system's performance against a hypothetical instantiation of DAISY, which does not incur any false positives or negatives in the threshold model (Optimal DAISY). Figure 9 shows the results for both accuracy and SLO attainment. We observe that the optimal DAISY model achieves slightly higher accuracy and SLO-attainment, due to its ability to make perfect decisions about model execution.

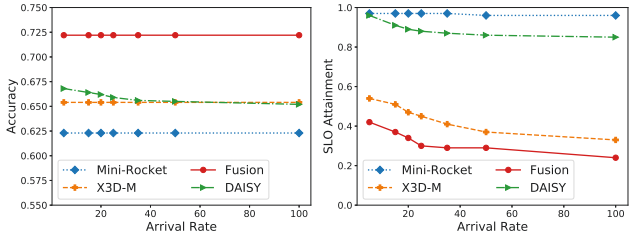
Our existing experiments assume we are able to leverage GPU resources when executing our threshold model to perform dynamic model selection for DAISY. However, it is possible that certain edge deployments are limited by GPU memory, which must be reserved for the inference tasks themselves. For this reason, we consider the performance of executing the DAISY model selection on CPU rather than GPU. Figure 10 shows the impact of CPU execution on accuracy and latency. We observe that executing model selection on CPU rather than GPU results in similar performance, with a very slight decrease in SLO attainment and accuracy. This technique still outperforms the Single-Modality approach, so it may be viable to execute model selection on the CPU if required.

Threshold vs Specialized Model. We now compare our two data-aware model selection strategies. We trained a specialized



(a) Accuracy (b) Inference Latency

Fig. 11: Comparison of a specialized and threshold model.



(a) Accuracy (b) SLO Attainment

Fig. 12: Single-Model Baselines

model (a slightly altered MiniRocket model) and compared it against the pre-trained MiniRocket model with a threshold. The objective was a binary classification problem, to determine whether to use m_{low} for inference or to defer to a more complex video model that could meet the latency SLO. Figure 11 shows a comparison between the two modeling options, where the red line represents the performance of the specialized model. For very high threshold values, a video model is executed for almost every request, which results in higher latency values. When the threshold is too low, the time series model is frequently used, even if it is not optimal. However, we observe that a properly configured threshold model can outperform a specially-trained model, both in terms of accuracy and overall inference latency (which includes the execution time of a secondary video model, if required). We conclude, based on theoretical and empirical findings, that the threshold model is the most appropriate in this setting.

F. Single-Model Baselines

Finally, we consider a system which selects a single model and uses it exclusively for inference. This removes any overhead for model selection and ensures the selected model remains in GPU memory at all times. Figure 12 shows the trade-offs with this approach. The larger models offer consistently higher accuracy, but have low SLO attainment and the lightweight time series model offers lower accuracy but high SLO attainment. DAISY leverages data-awareness to navigate this space: providing accuracy slightly higher than the X3D-M video model with an SLO-attainment closer to the time series model.

G. Discussion

Edge-cloud offloading was excluded from this analysis since some jurisdictions may require video data (especially patient data) to be processed locally to preserve privacy. However, our framework could easily be augmented to provide that support in less-restrictive environments. Privacy-preserving transformations could also be applied to images (e.g. blurring faces) at the edge site prior to offloading data to the cloud. This would require additional analysis to ensure the transformations do not degrade downstream inference accuracy.

VI. Related Work

A. Inference Serving in the Cloud

The InFaaS system introduced the notion of model-less inference serving, where inference requests arrive and the server is responsible for dynamically selecting a model to service the request [37]. The MArk system uses predictive autoscaling and serverless functions to provide consistent performance across various workloads [38]. The Clipper system performs model selection and uses dynamic batching to reduce inference latency [39]. Many of these cloud inference serving systems rely on homogeneous hardware and resource elasticity, both of which are unavailable at the edge. Furthermore, offloading data to the cloud may be infeasible due to constrained WAN links or data sovereignty / privacy requirements.

B. Video Analytics at the Edge

Other related works deal directly with challenges presented by video analytics and deep learning. Video analytics pipelines have many possible configurations, which can affect performance. Chameleon attempts to dynamically identify optimal configurations for each stream to improve efficiency [40]. It uses the heuristic that video cameras in close proximity may be *spatially dependent*, and will benefit from the same configuration. Ekya performs continuous learning and inference over multiple video streams at a single edge [1]. It proposes a novel scheduling algorithm that navigates the trade-off between inference accuracy and the potential accuracy improvement associated with retraining. Another method for reducing DNN computation at the edge is to transfer a subset of the video frames to the cloud for inference [14], [16], [17], [15]. VideoEdge maximizes query accuracy by identifying the best pipeline configuration across an edge-cloud hierarchy [17]. LayerCake is another system that balances the trade-off between edge and cloud inference [15]. Given a latency target, it seeks to maximize accuracy by scheduling the best available model that meets the SLA. While all of these approaches directly address the computational burden associated with video analytics, they focus exclusively on a single modality (RGB video frames) when multiple modalities are available. A few existing works consider dynamic model selection for a single modality at the edge [41], [42], [43]. A common strategy is to leverage edge-cloud offloading when the edge resources are insufficient for the workload [43], [15]. Offloading user data to the cloud may not always be feasible; however, these complimentary approaches can be integrated

into our model selection framework, since we only require an expected latency to perform model selection.

C. Edge Streaming Systems

Several systems have been architected to handle stream processing at the edge [44], [45], [46]. The EdgeWise system improved stream processing for the edge by redesigning the scheduler and allocating an appropriate number of worker threads based on the target hardware platform [44]. Another edge system builds data-awareness into their stream processing, which supports hundreds of parallel streams and data-driven decisions across multi-modal data [45]. These techniques, while effective, fail to directly address the computation bottlenecks associated with video streams and deep learning.

VII. Conclusion

Modern edge applications have access to data that spans multiple modalities. These diverse data can utilize a broader set of models to improve accuracy and reduce latency. We proposed DAISY, which consists of mechanisms for dynamically selecting an appropriate inference model based on the available data and the target SLO. We observed up to a 5% improvement in inference accuracy and a 60% improvement in SLO attainment using these approaches.

Acknowledgement

This research was supported in part by the NSF under grant CNS-1908566.

References

[1] R. Bhardwaj *et al.*, "Ekya: Continuous learning of video analytics models on edge compute servers," in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. Renton, WA: USENIX Association, Apr. 2022, pp. 119–135.

[2] Q. Zhang *et al.*, "Edge video analytics for public safety: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1675–1696, 2019.

[3] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu, "E-eyes: Device-free location-oriented activity identification using fine-grained wifi signatures," ser. MobiCom '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 617–628.

[4] J. Park, R. J. Javier, T. Moon, and Y. Kim, "Micro-doppler based classification of human aquatic activities via transfer learning of convolutional neural networks," *Sensors*, vol. 16, no. 12, 2016.

[5] M. Lu, Y. Hu, and X. Lu, "Driver action recognition using deformable and dilated faster r-cnn with optimized region proposals," *Applied Intelligence*, vol. 50, no. 4, pp. 1100–1111, 2020.

[6] M. Sun *et al.*, "Application of bluetooth low energy beacons and fog computing for smarter environments in emerging economies," in *Cloud Computing, Smart Grid and Innovative Frontiers in Telecommunications*. Cham: Springer International Publishing, 2020, pp. 101–110.

[7] E. G. Renart, J. Diaz-Montes, and M. Parashar, "Data-driven stream processing at the edge," in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, 2017, pp. 31–40.

[8] I. E. Olatunji and C.-H. Cheng, *Video Analytics for Visual Surveillance and Applications: An Overview and Survey*. Cham: Springer International Publishing, 2019, pp. 475–515.

[9] J. Sanchez, C. Neff, and H. Tabkhi, "Real-world graph convolution networks for action recognition in smart video surveillance," in *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, 2021, pp. 121–134.

[10] S. Jain *et al.*, "Spatula: Efficient cross-camera video analytics on large camera networks," in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, 2020, pp. 110–124.

[11] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," 2018.

[12] R. J. Wang, X. Li, and C. X. Ling, "Pelee: A real-time object detection system on mobile devices," in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.

[13] W. Niu *et al.*, "Achieving real-time execution of transformer-based large-scale models on mobile with compiler-aware neural architecture optimization," *CoRR*, vol. abs/2009.06823, 2020. [Online]. Available: https://arxiv.org/abs/2009.06823

[14] V. Nigade, L. Wang, and H. Bal, "Clownfish: Edge and cloud symbiosis for video stream analytics," in *2020 IEEE/ACM Symposium on Edge Computing (SEC)*, 2020, pp. 55–69.

[15] S. Ogden and T. Guo, "Layercake: Efficient inference serving with cloud and mobile resources," in *2023 23nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, 2023.

[16] X. Wang *et al.*, "Dynamic dnn model selection and inference off loading for video analytics with edge-cloud collaboration," in *Proceedings of the 32nd Workshop on Network and Operating Systems Support for Digital Audio and Video*. New York, NY, USA: Association for Computing Machinery, 2022, p. 64–70.

[17] C.-C. Hung *et al.*, "Videoedge: Processing camera streams using hierarchical clusters," in *ACM/IEEE Symposium on Edge Computing (SEC)*, October 2018.

[18] A. Padmanabhan, N. Agarwal, A. Iyer, G. Ananthanarayanan, Y. Shu, N. Karianakis, G. H. Xu, and R. Netravali, "Gemel: Model merging for Memory-Efficient, Real-Time video analytics at the edge," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. Boston, MA: USENIX Association, Apr. 2023, pp. 973–994.

[19] "Falls and fall prevention in the elderly," https://www.ncbi.nlm.nih.gov/books/NBK560761/.

[20] P. Climent-Pérez *et al.*, "A review on video-based active and assisted living technologies for automated lifelogging," *Expert Systems with Applications*, vol. 139, p. 112847, 2020.

[21] T. G. Stavropoulos *et al.*, "Iot wearable sensors and devices in elderly care: A literature review," *Sensors*, vol. 20, no. 10, 2020.

[22] G. Ananthanarayanan *et al.*, "Real-time video analytics: The killer app for edge computing," *Computer*, vol. 50, no. 10, pp. 58–67, 2017.

[23] F. Jurado Romero, E. Munoz Diaz, and D. Bousdar Ahmed, "Smartphone-based localization for passengers commuting in traffic hubs," *Sensors*, vol. 22, no. 19, 2022.

[24] V. Nair, M. Budhai, G. Olmschenk, W. H. Seiple, and Z. Zhu, "Assist: Personalized indoor navigation via multimodal sensors and high-level semantic information," in *Computer Vision – ECCV 2018 Workshops*. Cham: Springer International Publishing, 2019, pp. 128–143.

[25] D. Iskandaryan, F. Ramos, and S. Trilles, "Air quality prediction in smart cities using machine learning technologies based on sensor data: A review," *Applied Sciences*, vol. 10, no. 7, 2020.

[26] M. E. Ilas and C. Ilas, "Towards real-time and real-life image classification and detection using cnn: a review of practical applications requirements, algorithms, hardware and current trends," in *2020 IEEE 26th International Symposium for Design and Technology in Electronic Packaging (SIITME)*, 2020, pp. 225–233.

[27] C. Canel *et al.*, "Scaling video analytics on constrained edge nodes," in *Proceedings of Machine Learning and Systems*, vol. 1, 2019, pp. 406–417.

[28] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[29] "Internet of things (iot) connected devices installed base worldwide from 2015 to 2025(in billions)," https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[30] S. Yi, Z. Hao, Q. Zhang, Q. Zhang, W. Shi, and Q. Li, "Lavea: Latency-aware video analytics on edge computing platform," ser. SEC '17. New York, NY, USA: Association for Computing Machinery, 2017.

[31] H. Choi, A. Beedu, H. Haresamudram, and I. Essa, "Multi-stage based feature fusion of multi-modal data for human activity recognition," 2022.

[32] M. M. Islam and T. Iqbal, "Mumu: Cooperative multitask learning-based guided multimodal fusion," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 1, pp. 1043–1051, Jun. 2022.

[33] Q. Kong, Z. Wu, Z. Deng, M. Klinkigt, B. Tong, and T. Murakami, "Mmact: A large-scale dataset for cross modal human action understanding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[34] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2020, pp. 200–210.

[35] A. Dempster, D. F. Schmidt, and G. I. Webb, "Minirocket: A very fast (almost) deterministic transform for time series classification," ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 248–257.

[36] W. Seo, S. Cha, Y. Kim, J. Huh, and J. Park, "Slo-aware inference scheduler for heterogeneous processors in edge platforms," *ACM Trans. Archit. Code Optim.*, vol. 18, no. 4, jul 2021.

[37] F. Romero, Q. Li, N. J. Yadwadkar, and C. Kozyrakis, "INFaaS: Automated model-less inference serving," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. USENIX Association, Jul. 2021, pp. 397–411.

[38] C. Zhang, M. Yu, W. Wang, and F. Yan, "MArk: Exploiting cloud services for Cost-Effective, SLO-Aware machine learning inference serving," in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. Renton, WA: USENIX Association, Jul. 2019, pp. 1049–1062.

[39] D. Crankshaw, X. Wang, G. Zhou, M. J. Franklin, J. E. Gonzalez, and I. Stoica, "Clipper: A Low-Latency online prediction serving system," in *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. Boston, MA: USENIX Association, Mar. 2017, pp. 613–627.

[40] J. Jiang *et al.*, "Chameleon: Scalable adaptation of video analytics," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, ser. SIGCOMM '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 253–266.

[41] V. S. Marco, B. Taylor, Z. Wang, and Y. Elkhatib, "Optimizing deep learning inference on embedded systems through adaptive model selection," *ACM Trans. Embed. Comput. Syst.*, vol. 19, no. 1, feb 2020.

[42] K. Zhao, Z. Zhou, X. Chen, R. Zhou, X. Zhang, S. Yu, and D. Wu, "Edgeadaptor: Online configuration adaption, model selection and resource provisioning for edge dnn inference serving at scale," *IEEE Transactions on Mobile Computing*, vol. 22, no. 10, pp. 5870–5886, 2023.

[43] X. Wang, G. Gao, X. Wu, Y. Lyu, and W. Wu, "Dynamic dnn model selection and inference off loading for video analytics with edge-cloud collaboration," in *Proceedings of the 32nd Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 64–70.

[44] X. Fu *et al.*, "EdgeWise: A better stream processing engine for the edge," in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. Renton, WA: USENIX Association, Jul. 2019, pp. 929–946.

[45] E. G. Renart, J. Diaz-Montes, and M. Parashar, "Data-driven stream processing at the edge," in *2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*, 2017, pp. 31–40.

[46] J. de Oliveira *et al.*, "Knowledge graph stream processing at the edge," in *Proceedings of the 16th ACM International Conference on Distributed and Event-Based Systems*, ser. DEBS '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 115–125.