# AdaFlowLite: Scalable and Non-blocking Inference on Asynchronous Mobile Data

Sicong Liu, *Member, IEEE,* Fengmin Wu, Yuan Gao, Bin Guo*, *Senior Member, IEEE,* Zimu Zhou, *Member, IEEE,* Hongkai Wen, *Member, IEEE,* Zhiwen Yu, *Senior Member, IEEE*

***Abstract*—The rise of mobile devices equipped with numerous sensors, such as LiDAR and cameras, has driven the adoption of multi-modal deep intelligence for distributed sensing tasks, such as smart cabins and driving assistance. However, the arrival time of mobile sensory data vary due to modality size and network dynamics, which can lead to delays (if waiting for slow data) or accuracy decline (if inference proceeds without waiting). Moreover, the diversity and dynamic nature of mobile systems exacerbate this challenge. In response, we present a shift to *opportunistic* inference for asynchronous distributed multi-modal data, enabling inference as soon as partial data arrives. While existing methods focus on optimizing modality consistency and complementarity, known as modal affinity, they lack a *computational* approach to control this affinity in open-world mobile environments. AdaFlowLite pioneers the formulation of structured cross-modality affinity in mobile contexts using a hierarchical analysis-based normalized matrix. This approach accommodates the diversity and dynamics of modalities, generalizing across different types and numbers of inputs. Employing an multi-modal lightweight Swin Transformer (MMLST), AdaFlowLite facilitates real-time and flexible data imputation, adapting to various modalities and downstream tasks without retraining. Experiments show that AdaFlowLite significantly reduces inference latency by up to 80.4% and enhances accuracy by up to 62.1%, while achieving nearly a 50% reduction in energy consumption, outperforming status quo approaches. Also, this method can enhance LLM performance to preprocess asynchronous data.**

***Index Terms*—Mobile context, Multi-modality Inference, Modality Affinity Control.**

## I. Introduction

The broad integration of low-power, rich-data sensors like cameras, LiDAR, acoustic sensors, hyperspectral cameras, and radio frequency detectors has greatly advanced the sensing potential of mobile devices [1], [2]. Leveraging multiple sensors for multi-modal inference provides robust perception outcomes compared to single-sensor systems across various domains, such as smart cockpits (*e.g.* Google automotive service [3]), driving assistance (*e.g.* Tesla Autopilot [4]), 3D scene understanding (*e.g.* Microsoft Azure Kinect DK [5]), health monitoring (*e.g.* Apple HealthKit [6]), and AR/VR (*e.g.* Meta Oculus Rift [7]). The advantages of *distributed* multi-modal systems are particularly evident in challenging environments or objects. For instance, observing occluded objects, penetrating through walls, operating in low-light conditions, and navigating around corners. The diversity of sensor data from various fields of view and sensitivities supports independent cross-validation from multiple vantage points and enables detailed extraction of complex patterns (*e.g.* location, direction, and materials).

*Corresponding Author: Bin Guo (e-mail:guob@nwpu.edu.cn)

In mobile and distributed computing systems, a fundamental challenge lies in efficiently processing and analyzing massive real-time multi-modal data streams generated by advanced sensors, especially in vehicle-/drone-mounted scenarios (we defer more detail in Sec. IV-A). Many studies focus on technologies and tools for handling multi-modal data, such as adopting AdaptSegNet for fusing complex signals [8], or further explored partitioning models across mobile devices and the cloud to enhance processing speed [9]. However, the significant *gap* exists in open-world mobile deployments, characterized by input *asynchrony* and *heterogeneity*:

• **Asynchronism**. In distributed multi-modal systems, particularly those employing vehicle-/drone-mounted sensors, the diversity of input modalities (*e.g.* video *vs.* LiDAR) coupled with dynamic network conditions often leads to asynchronous data arrivals at the fusion device. These challenges are further compounded by the resource constraints typical of mobile systems, such as limited computational power and the absence of clock synchronization among low-power mobile sensors. These factors can result in delays when the system waits for slower data streams or a decline in accuracy if fusion proceeds without these data streams being synchronized. Consider a real-world example involving an autonomous driving system equipped with six cameras and one LiDAR sensor. In this scenario, an asynchronous delay of approximately 40ms may occur for LiDAR data over a 100Mbps bandwidth, given a data size ratio of $1:4$ between LiDAR and camera data. This delay can increase the total latency from 1s to 4.3s for processing 81 frames within a $4s$ window when the system chooses to wait for the slower LiDAR data. Alternatively, proceeding without waiting for data synchronization can degrade the fusion accuracy by 49.7%. To maintain operational safety and efficiency, mobile systems must minimize latency, such as $\leq$ 26ms per frame [10]. We defer more details to Sec. II-A1.

• **Heterogeneity**. In distributed multi-modal systems, particularly those with vehicle-/drone-mounted sensors, variations in the field of view, sensitivity, noise (e.g., lighting conditions), and feature distribution significantly influence the *consistency* and *complementarity* of data fusion outcomes [11]. The effective management of these factors is crucial when deciding whether to discard or impute slow-arriving modalities, as these decisions directly impact inference accuracy and system latency. For instance, we find that discarding 50% of the slow data, which exhibits approximately 33% arrival delays, results in a 28.1% decrease in accuracy using the real-world autonomous driving dataset, nuScenes [12]. Employing the K-Nearest Neighbors (KNN) approach [13], a common method for asynchronous data imputation for addressing this problem still leads to a 9.7% reduction in accuracy and a 328% increase in latency. Additional details on this observation are presented in Sec. II-B.

As a separate note, cross-modal *consistency* and *complementarity*, key components of what we describe modality ***affinity***, are crucial for enhancing inference robustness and accuracy by leveraging *modality-shared* and *modality-specific* information, respectively [14]. In diverse mobile scenarios, the addition of sensors necessitates a *refined cross-modality affinity analysis*. Current affinity analysis often relies on *complex manual* adjustments and significant computational overhead, as detailed in Sec. II-A2. Moreover, modality affinity is subject to fluctuations due to input *asynchronism* and *heterogeneity*, which vary not only across tasks but also with identical inputs, as observed in Sec. II-B. Considering these characteristics, both existing synchronous (blocking) methods [1], [15], [16] and asynchronous (non-blocking) approaches [17], [18], [19] encounter significant challenges as below:

• ***Challenge #1***: Most multi-modal methods balance accuracy and latency by enhancing data *consistency* and *complementarity*. However, these methods often only provide qualitative affinity analysis under ideal conditions [20], [21]. And both qualitative or quantitative method(*i.e.* the preliminary AdaFlow [19]), are primarily designed to handle the *reduction* (*i.e.* missing subsets) of modalities but fail to accommodate the *addition* of sensor types or numbers. Our approach utilizes a *prior-free modality affinity refinement* that quantitatively analyzes cross-modal affinity and scaly accommodate to new modalities. This facilitates the effective management of diverse and dynamic inputs in systems equipped with vehicle-/drone-mounted sensors.

• ***Challenge #2***: Scheduling asynchronous and heterogeneous multi-modal data for low-latency inference on resource-constrained mobile platforms such as vehicle-/drone-mounted systems is non-trivial. It involves NP-hard multi-choice, multi-strategy optimization problems [20], [22], [23]. Resource limitations inherent to mobile environments make the task even more challenging. Moreover, designing a lightweight one-fit-all data imputation module that can handle *dynamic* and *diverse* input modalities, various subsequent tasks, and fluctuating asynchronous delays is essential yet challenging.

To address the challenges of asynchronous data integration in mobile environments with resource constraints, we propose a shift to *opportunistic* inference. The server (*e.g.* on a vehicle-/drone-mounted mobile device) initiates inference processes as soon as any partial data becomes available, optimizing the use of limited computational resources. This method is particularly effective in scenarios where data collection is subject to variable delays and interruptions, common in mobile settings. AdaFlowLite leverages fully computational *modality affinity* to ensure robust data fusion at runtime, that not only models but also efficiently optimizes inference activities based on the affinity between different data modalities.

• ***First***, inspired by [20], which investigates *task* affinity using *unimodal* (visual) data during *learning*, we extend this to the *modality* level during *inference*, enabling dynamic adaptation across diverse modalities. AdaFlowLite leverages t-SNE combined with the Analytic Hierarchy Process (AHP) to normalize the modality affinity matrix. To not only handle *missing* modalities but also incorporate new *unseen* sensors/modalities, AdaFlowLite integrates a *prior-free affinity refinement* module, enabling seamless integration into the structured modality affinity matrix. This ensures robust decision-making in dynamic data availability and diverse mobile scenarios.

• ***Second***, to enable low-latency non-blocking inference on resource-limited devices that adapts to varying input modalities and types *without retraining* at runtime, we introduce a *one-fit-all multi-modal lightweight Swin Transformer* (MMLST). It adaptively processes diverse input features and modalities and finely controls modal affinity in dynamic opportunistic inference. Notably, the proposed MMLST is *lightweight* and suitable for resource-constrained mobile devices such as vehicle-/drone-mounted systems.

We evaluate the performance of AdaFlowLite on real-world 3D object recognition tasks and scenarios with diverse data heterogeneity and asynchrony. Results show a reduction of up to 80.4% in inference time with an improvement of up to 62.1% in accuracy, while achieving nearly a 50% reduction in energy consumption. Especially, it displays the affinity matrix enables opportunistic inference based on different downstream tasks and resource-constrained scenarios. This method can also be a pre-module for LLM, enhancing performance on asynchronous data. Our main contributions are as follows.

- As far as we know, this is the first work to integrate quantitative modality affinity control into distributed multi-modal inference, addressing *system asynchrony* and *diverse, dynamic demands*. It eliminates waiting times for slow data and minimizes accuracy loss when excluding such data.
- We introduce AdaFlowLite, a system for opportunistic inference that adapts to asynchronous and heterogeneous data streams. It harnesses structured and scalable affinity control to enable adaptive fusion of dynamic data. Also, AdaFlowLite proposes a lightweight MMLST to achieve precise and timely data imputation on mobile devices.
- Experiments show that AdaFlowLite outperforms existing a-/synchronous methods [24], [25], [26], [13], [19] in trading off between inference accuracy, latency and energy consumption across various real-world mobile tasks, inputs, and scenarios.

## II. Overview

### A. Problem Analysis

*1) Data Asynchronism Problems:* To analyze and observe the motivation, we conducted the following experimental tests. We test a distributed multi-modal 3D object recognition system on nuScenses dataset [12]. nuScenses, collected from real-world autonomous driving scenarios, includes 6 camera and 1 LiDAR views. Given the 1:4 data size ratio between LiDAR and camera data, a 40ms asynchronous delay occurs in LiDAR data on a 100Mbps bandwidth, a common scenario in vehicle-mounted and drone-mounted applications where bandwidth is limited and data throughput must be optimized. *First*, as illustrated in Fig. 1a, we simulate data *asynchronism* using diverse missing rates of slow modality (*i.e.* 3D LiDAR point cloud), *i.e.* 0%, 25%, 50%, and 75%. As the missing rate of the slow modality increases, while the fast modality (*i.e.* RGB images) remains fully available, if not waiting for slow data, the inference accuracy progressively declines (*e.g.* 49.8%). If waiting for slow data, the latency increases from 1s to 4.3s for 81 frames. *Second*, as shown in Fig. 1b, we test various imputation methods for *asynchronous fusion* such as SPC (Sparse Point Cloud) [25], KNN (K-Nearest Neighbors) [13], and PCN (Point Completion Net) [26], along with the *synchronous* fusion method BM (Blocking Mechanism) [24], across missing rates of 0%, 25%, 50%, and 75% in the slow data (*i.e.* 3D point cloud) when the fast data (*i.e.* RGB image) is available. We find that while the blocking method offers the highest accuracy, its latency, nearly equal to the sampling interval of the test data (4s), is too high. Conversely, the sparse point cloud method is the fastest but yields an unacceptable accuracy drop (*i.e.* 4.29%).

*2) Sensor Scalability Problem:* Multi-modal systems commonly *assume* that the total number of sensors remains *constant* during operation, a foundational premise in existing frameworks [1]. However, in diverse mobile scenarios where sensors need to be *added*, whether to refine sensing domains or enhance capabilities, integrating *new* and potentially *unseen* sensors becomes a significant challenge. This process requires *refining the cross-modality affinity analysis* to accurately establish relationships between new and existing sensors/modalities for effective fusion. However, current affinity analysis often relies on *complex manual* derivations and computational overhead [27], making it both time-consuming and impractical for non-experts to implement. The challenge intensifies in scenarios such as foldable or flexible grippers [28], where dynamic, real-time modality affinity analysis is essential but remains highly challenging. Existing methods, whether qualitative [21] or quantitative (*i.e.* the preliminary AdaFlow [19]), are primarily designed to handle the *reduction* (*i.e.* missing subsets) of modalities but fail to accommodate the *addition* of sensor types or numbers. This limitation exposes a critical gap in enabling *scalable* sensor and modality integration (*i.e.* both scaling up and down), an essential capability for diverse multi-modal systems.

### B. Observations and Opportunities

In mobile applications, especially in resource-constrained environments like vehicle-/drone-mounted systems, real-time distributed multi-modal data fusion is critical for balancing inference accuracy and latency. A key challenge of existing methods (discussed in Sec. VI) is their inability to quantitatively assess and manage *modality affinity*, which encompasses both *consistency* and *complementarity* across modalities. This gap often results in delays or accuracy drops, exacerbated by the inherent asynchrony and heterogeneity of mobile contexts. Despite its importance, modality affinity remains a *black box*.

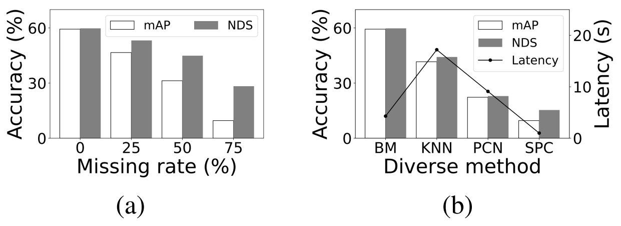


Fig. 1. (a) Inference accuracy, (b) accuracy & latency of existing syn-/asynchronous methods, under varying missing rates of slow modality (*i.e.* 3D point cloud).

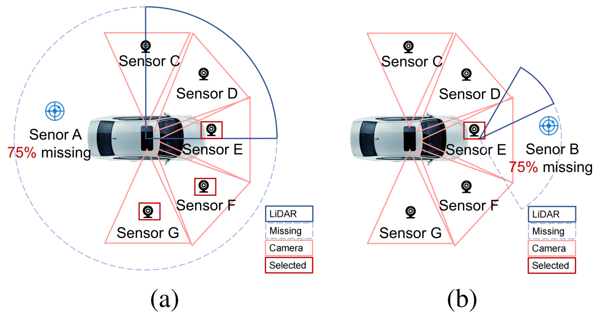


Fig. 2. (a) Top LiDAR emphasizes complementarity and (b) Non-Top LiDAR emphasizes consistency.

*1) Modality-specific Modality Affinity:* Different sensors on mobile devices exhibit distinct modality affinities due to their varying view and size, affecting how they interact with data from other modalities, emphasizing consistency and complementarity. For instance, consider a vehicle equipped with seven sensors including top and front LiDARs (Sensor $A$ and $B$), and five positioned cameras (sensor C$\sim$G). As illustrated in Fig. 2a, when the top full-view LiDAR sensor $A$ (slow data with larger size) misses 75% of its data, selecting data from sensors C$\sim$G enhances *complementarity* and broadens perspectives, thus improving inference accuracy (*i.e.* from 9.6% to 54.9%). Conversely, Fig. 2b shows that when the front partial-view LiDAR lags 75% of its data due to network issues, prioritizing data from sensors C$\sim$G for greater *consistency*, rather than *complementarity*, yields better accuracy. This is because the partial-view LiDAR provides specific directional data, necessitating supplementary directional data from other cameras to maintain noise-robust results. This indicates that the types and number of available fast modalities change dynamically in asynchronous systems. Consequently, modality selection for fusion must adapt dynamically to maintain an optimal performance(the necessity analysis discussed in IV-B and the experimental result is in V-F1).

*2) Subsequent Task-specific Modality Affinity:* . Even the same multi-modal sensor arrival patterns can exhibit varying modality affinities across different *downstream tasks*. As demonstrated in Fig. 4, we adopt the same heterogeneity and asynchrony settings, *i.e.* 70ms delays of each frame, corresponding to 50% missing rate, to fairly test three diverse downstream tasks: *i) BEV object detection (BOD)*, which involves object recognition from a bird's eye view; ii) *Bounding box segmentation (BBS)*, which uses a rectangular box to represent

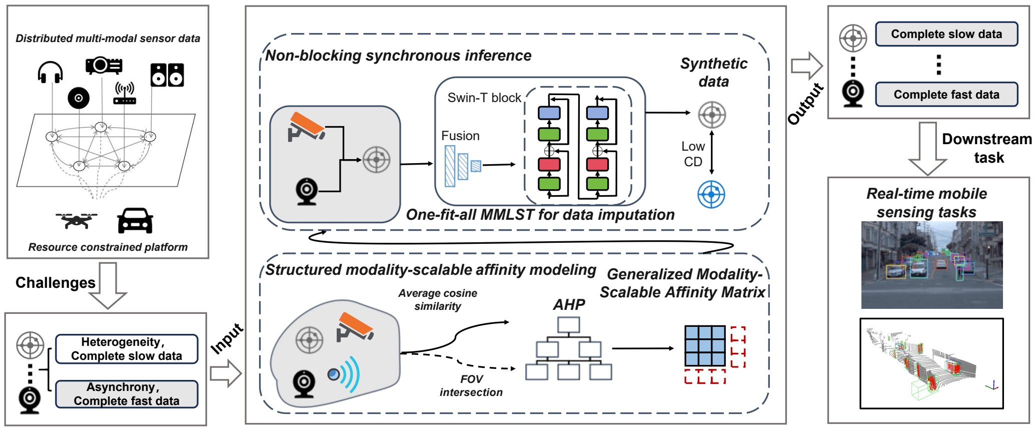

Distributed multi-modal sensor data
Resource constrained platform
Challenges
Heterogeneity, Complete slow data
Asynchrony, Complete fast data
Input
Non-blocking synchronous inference
Swin-T block
Fusion
Synthetic data
Low CD
One-fit-all MMLST for data imputation
Structured modality-scalable affinity modeling
Average cosine similarity
FOV intersection
AHP
Generalized Modality-Scalable Affinity Matrix
Output
Complete slow data
Complete fast data
Downstream task
Real-time mobile sensing tasks

Fig. 3. Workflow of AdaFlowLite.

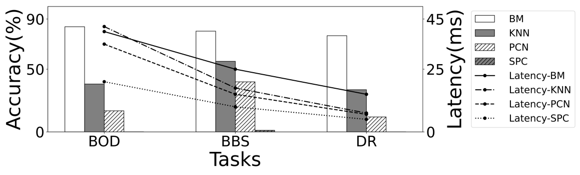


Fig. 4. Inference latency and accuracy of BOD, BBS, DR tasks in existing a-/synchronous methods.

target objects in images or point clouds for semantic segmentation; *iii) Direction recognition (DR)*, which compares direction similarity between the target and the actual orientation. The results show that employing a naive synchronous method (*i.e.* BM [24]) results in significant variations in latency across three tasks, *e.g.* 40ms for BOD, 25ms for BSS, and 15ms for DR task. This variance is due to the distinct affinity criteria shaped by downstream tasks, leading to disparities in inference latency especially when data arrives synchronously. Such variability is prohibitive in latency-sensitive applications. Moreover, different data imputation techniques for asynchronous fusion (*i.e.* SPC [25], KNN [13], PCN [26]) result in even more pronounced variations across three tasks. For example, when using PCN asychronous method, the inference latency across three tasks are 35ms, 15ms, 7ms, respectively. While accuracy are 16.8%, 39.9%, and 11.9%, respectively.

These observations demonstrate that *dynamics* in mobile sensing patterns, driven by modality asynchrony, heterogeneity, or subsequent tasks, desired *generalized* modality affinity criteria. Building this criteria has notable values.

## C. Solution Overview

To address the above challenges in mobile distributed multi-modal inference systems, we introduce AdaFlowLite (Sec. I). Fig. 3 shows the architecture of AdaFlowLite (Fig. 17 shows the idea of AdaFlowLite), which includes two key modules: Structured Modality-Scalable Affinity Modeling and Lightweight Non-blocking Asynchronous Inference. *First*, the Structured Modality-Scalable Affinity Modeling module (Sec. III), central to AdaFlowLite, tackles *Challenge #1* by constructing a *generalized* modality-scalable affinity matrix. This matrix is specifically designed to manage the inherent *asynchrony* and *heterogeneity* of different modalities, which are common in complex sensing environments. Furthermore, AdaFlowLite incorporates a prior-free affinity refinement module for seamlessly integrating new sensors into the structured modality affinity matrix by utilizing *prior-free analysis*, a critical feature in diverse and dynamic mobile settings where prior knowledge of sensor characteristics is often unavailable. Also, these capabilities are crucial for dynamically assessing input modalities, particularly in resource-constrained platforms such as vehicles or drones, ensuring robust performance and adaptability under diverse mobile scenarios. *Second*, the Lightweight Non-blocking Asynchronous Inference module (Sec. IV) addresses *Challenge #2* by introducing the multi-modal lightweight Swin Transformer (MMLST). This model adapts to various input/output modalities without retraining at runtime, ensuring real-time, accurate inference through dynamic modality fusion. It also uses a lightweight dynamic programming approach to select optimal affinity sub-graphs from the matrix, enhancing performance in mobile systems with limited resources.

## III. Structured Modality-Scalable Affinity Modeling

### A. Preliminary of Generalized Cross-modality Affinity Matrix

In mobile scenarios characterized by the diversity and dynamics of distributed multi-modal data, which often vary in views and arrival times, ensuring robust decision-making under fluctuating data availability and diverse conditions presents significant challenges. A *generalized modality affinity matrix* is essential to address these challenges by managing the *heterogeneity* and *asynchrony* of data. This matrix enhances AdaFlowLite's ability to capture complex, nonlinear relationships in high-dimensional data, thereby accommodating diverse tasks alongside dynamically changing input data modalities that are non-stationary, *asynchronous*, and *heterogeneous*.

In a preliminary version of AdaFlowLite, namely AdaFlow [19], we introduced a mathematical framework called the ***affinity matrix***. This framework is designed to tackle the challenges of modality heterogeneity, asynchrony, and varied views within mobile scenarios through a structured two-step strategy.

In the first step, AdaFlow employs the t-SNE method [29], which maps Euclidean distances between high-dimensional data points into joint probabilities that reflect their similarities. This process projects heterogeneous modalities into a high-dimensional feature space and excels at information-lossless dimensionality reduction, essential for preserving data integrity. Subsequent to this reduction, AdaFlow calculates pairwise modalities' affinity, ensuring accurate modality fusion. Although obtaining pair-wise modalities' affinity using t-SNE is a straightforward procedure, applying this algorithm globally (across diverse and dynamic mobile setting) and incorporating every single pair-wise modalities' affinity into a unified framework transforms the problem into a NP-hard problem.

In the second step, AdaFlow addresses the diversity and dynamics of multi-modal fusion by utilizing the ***Analytic Hierarchy Process (AHP)***. AdaFlow are not aiming for a globally optimal solution. Instead, it starts with slow data for planning, which reduces the NP-hard problem to a multi-objective programming problem that is more suitable for AHP. The affinity matrix in AdaFlow has three layers: the *Focus*, *Criteria*, and *Alternative* layers. The *Focus* layer prioritizes sensors with lower data generation rates or higher vulnerability to lagging, while the *Alternatives* layer includes sensors that transmit data to the terminal first. The *Criteria* layer considers two main factors: the average cosine similarity from the t-SNE visualization and the Field Of View (FOV) intersection between different sensors.

To illustrate how an affinity matrix is established, there is a simple example. The process involves two distinct matrices, A and $\{B_1, B_2\}$, for normalizing affinity values. Within matrix A, the element $a_{ij}$ represents the relative importance of each pairwise criterion contrast in the criterion layer. The scaling method for $a_{ij}$ is rule-based. In this example, matrix A is straightforward, consisting of two rows and two columns. $C_1$ comes from t-SNE method and $C_2$ comes from the intersection on the Field Of View (FOV). The example assigns $a_{12} = 7$ and $a_{21} = \frac{1}{7}$.

$$A = \begin{array}{c} \\ C_1 \\ C_2 \end{array} \begin{array}{c} \begin{array}{cc} C_1 & C_2 \end{array} \\ \begin{bmatrix} 1 & 7 \\ \frac{1}{7} & 1 \end{bmatrix} \end{array}$$

In the matrix $B_i$, the element $b_{ij}$ represents the importance of sensor $i$ relative to sensor $j$ under criterion $C_i$. This importance is derived by standardizing the average cosine or FOV values of each sensor in t-SNE. For example, with one Focus sensor and three Alternative sensors $\{sensor_1, sensor_2, sensor_3\}$, the matrix $B_i$ is a $3 \times 3$ matrix.

$$B_1 = \begin{bmatrix} 1 & 2 & 5 \\ \frac{1}{2} & 1 & 2 \\ \frac{1}{5} & \frac{1}{2} & 1 \end{bmatrix} \quad B_2 = \begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{8} \\ 3 & 1 & \frac{1}{3} \\ 8 & 3 & 1 \end{bmatrix}$$

where $A$ and $\{B_1, B_2\}$ all pass the consistency test, their eigenvalues can be approximated by the arithmetic mean of their column vectors, thus obtaining the eigenvectors.

$$W_1 = \begin{bmatrix} 0.875 \\ 0.125 \end{bmatrix} \quad W_2 = \begin{array}{c} \begin{array}{cc} B_1 & B_2 \end{array} \\ \begin{bmatrix} 0.594 & 0.082 \\ 0.277 & 0.236 \\ 0.129 & 0.682 \end{bmatrix} \end{array}$$

Then,

$$W = W_1 \cdot W_2 = \begin{bmatrix} 0.53 \\ 0.272 \\ 0.198 \end{bmatrix}$$

According to values in $W$, for the Focus sensor, $B_1$ has the greatest affinity, followed by $B_2$, with $B_3$ being the least. This process yields a vector $W$ for each sensor. These vectors are assembled to form the final affinity matrix, as depicted in Fig. 5a, which is crucial for robust decision-making in dynamic environments. The outcome of this two-step strategy is a generalized affinity matrix. Please refer to [19] for more detailed formula derivation and theoretical proof.

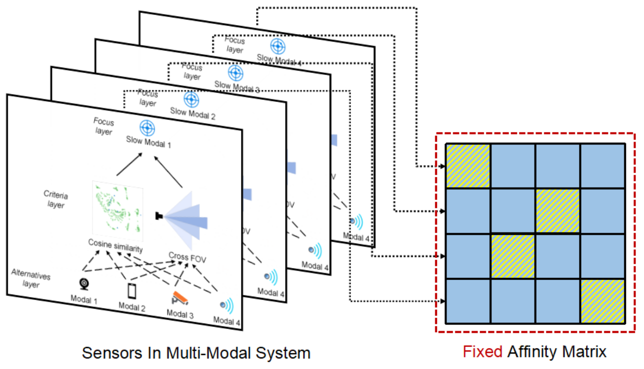
## B. Prior-free Modality Affinity Refinement

**Limitations of AdaFlow's Affinity Matrix**. AdaFlow's affinity matrix faces significant challenges in diverse mobile scenarios where new sensors without prior knowledge are integrated to enhance the multi-modal system's perception capabilities or refine local perception areas. This limitation primarily stems from the inherent design of the affinity matrix, which, as detailed in Sec. III-A, relies on dimensionality reduction techniques to establish affinities between sensors and their fields of view (FOV). Among them, FOV presuppose the availability of prior knowledge, thus constraining the matrix's functionality when encountering *unseen* modalities.

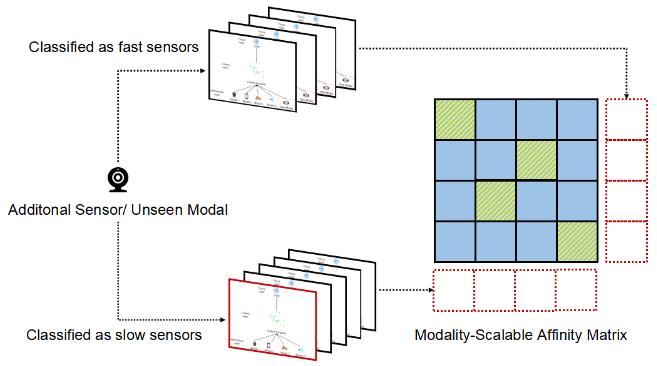
Furthermore, as illustrated in Fig. 5a, the structure of AdaFlow's affinity matrix is fixed upon construction. Introducing new sensors into this established matrix can lead to passive deformations, compromising the system's stability. To address these challenges, a scalable approach is desired to allow the affinity matrix to actively transform in response to the addition of new sensors, thereby maintaining stability and ensuring continuous system adaptability.

**Design**. To effectively manage *unseen* modalities in our design, we propose a *prior-free analysis* based on AdaFlow's affinity matrix. Specifically, in scenarios where sensors need to be *added*, we only utilize t-SNE exclusively as the criterion. This method does not strictly require historical data but instead focuses on the features of current sensor data. This simplification allows us to bypass the necessity for prior knowledge, particularly concerning the field of view (FOV), which plays a minor, auxiliary role in AdaFlow's affinity matrix. By removing this component, we accept a trade-off in output accuracy for greater flexibility in integrating new sensor types without compromising the overall system functionality.

In our preliminary design, AdaFlow's affinity matrix categorizes sensor modalities into two groups: fast and slow, based on their asynchronous data arrival times, which are influenced



(a) AdaFlow's fixed affinity matrix


Classified as fast sensors
Additonal Sensor/ Unseen Modal
Classified as slow sensors
Modality-Scalable Affinity Matrix

(b) AdaFlowLite's modality-scalable affinity matrix

Fig. 5. (a) AdaFlow's fixed affinity matrix, (b) AdaFlowLite's modality-scalable affinity matrix.

by factors such as data volume and network bandwidth. This categorization also aids in reducing computational complexity during affinity refinement. Building upon this foundation, we introduce the structured refinement strategy to the modality affinity matrix. Specifically, rows correspond to slow sensors, while columns represent fast sensors. This arrangement ensures matrix stability and responsiveness to dynamic changes in the sensor landscape. It enables seamless adaptation to new sensor modalities without incurring *manual complexity* or significant computational overhead, making the approach both scalable and efficient.

• **Slow Sensors**. The integration of a newly classified slow sensor into AdaFlowLite's modality affinity matrix is visually represented by the addition of a new row, as illustrated in Fig. 5b. This addition reflects the affinity vector corresponding to the slow sensor. Given that each row in the matrix represents independent affinity vectors, as illustrated in Fig. 5a, the introduction of a new sensor does not necessitate recalculating existing vectors, thereby preserving computational resources. In our *prior-free analysis*, the matrix groups $\{B_1, B_2\}$ are simplified into a single matrix $B_1{}'$. This new matrix maintains its original structure but is now populated with standardized affinity scores between the existing slow sensors and the newly added one. Meanwhile, matrix A is simplified to a unit matrix, this refinement reducing complexity and focusing the affinity analysis on the new interactions. As a result, the affinity vector for the new sensor, $W = W_2{}'$, is directly derived from the eigenvectors of $B_1{}'$, and AdaFlowLite's modality affinity matrix will formed as below:

$$\begin{pmatrix} {W^1}^T \\ \vdots \\ {W^i}^T \\ {W^{i+1}}^T \end{pmatrix}$$

• **Fast Sensors**. The integration of a newly classified fast sensor into the modality affinity matrix AdaFlowLite's is visually represented by the addition of a new column, as illustrated in Fig. 5b. This process contrasts with the integration of slow sensors, which only requires adding a new row without necessitating recalculations of existing vectors. However, the introduction of fast sensors entails a more complex adjustment. These sensors influence every affinity vector within the matrix. Consequently, matrix $B_1{}'$, which is a simplified form matrix groups $\{B_1, B_2\}$, undergoes a transformation to accommodate the new sensor dynamics. This necessitates recalculating the affinity vectors to reflect the updated sensor interactions, ensuring that the matrix accurately represents the enhanced sensing capabilities and maintains system integrity. $B_1{}'$ will transformed as below:

$$B_1{}' = \begin{bmatrix} B_1 & b_{1j} \\ & \vdots \\ b_{i1} \cdots & 1 \end{bmatrix}$$

The rest of the calculation are the same as for *slow sensors*, and AdaFlowLite's modality affinity matrix will formed as below:

$$\begin{pmatrix} {W^1}'^T \\ \vdots \\ {W^i}'^T \end{pmatrix}$$

This refinement ensures that the matrix remains adaptable and efficient, capable of scaling to accommodate new sensors without significant recalculations or overhead.

## IV. Lightweight Non-blocking Asynchronous Inference

This subsection then explores the adaptive fusion of asynchronous and heterogeneous input via imputation, ensuring non-blocking inference without compromising accuracy and maintaining compute consumption low. In a preliminary version of AdaFlowLite, we used a CNN-based GAN for one-fit-all data imputation as in AdaFlow [19]. However, for complex data like point clouds, CNN-based GANs require deeper architectures and larger parameters to capture data correlations, which can hinder real-time performance on resource-limited mobile platforms. In contrast, the Transformer's scalability and parallel processing capabilities make it a more efficient and lightweight alternative for AdaFlowLite, enabling one-fit-all data imputation without sacrificing performance.

## A. Lightweight One-fit-all Data Imputation

**Why Transformer**. The Transformer architecture has strong applicability across diverse *data types* and *tasks*, showing particular promise for data imputation for three key reasons: *i)* The self-attention mechanism enables Transformers to effectively capture long-range dependencies in data streams, making them well-suited for mobile data imputation [30]. *ii)* Transformers can be scaled (in terms of layers, heads, *etc.*) to suit varying task complexities, offering flexibility for optimizing dynamic data imputation tasks [31]. *iii)* The architecture allows for parallel processing of entire sequences, improving efficiency, critical for real-time performance demands on resource-constrained mobile devices with complex input [32].

**Limitations of Prior Arts**. While directly using Vanilla Transformer [30] poses challenges in parameter storage, we adopt the Swin Transformer [33] for its computational efficiency. The Swin Transformer reduces complexity by restricting self-attention calculations to localized windows, which also improves scalability. Additionally, its *Shifted Window* mechanism enhances inter-window communication, boosting the model's expressive power. This localized attention strategy is particularly well-suited for imputation tasks, making the Swin Transformer a highly effective choice for handling such generalized and dynamic data imputation requirements on mobile devices. However, due to *dynamic asynchrony* and *heterogeneity* in mobile sensing patterns, directly applying Swin Transformer to a distributed multi-modal system is often ineffective. This is because the traditional Swin Transformer typically assumes *synchronous* inputs with *fixed* input volume and type, which does not always hold in real-world mobile distributed systems. For example, slow LiDAR data with varying sensing patterns requires video data from different cameras for imputation.

**Our Scope**. To tackle these challenges, we extend the Swin Transformer to multi-modal settings and integrate it with the proposed affinity matrix using a lightweight architecture, namely the Multi-modal Lightweight Swin Transformer (MMLST). The Swin Transformer captures correlations between fast and slow data modalities (*e.g.* RGB images and point clouds) by utilizing attention windows of varying sizes. The architecture of the MMLST is shown in Fig. 7. The MMLST architecture has two key components: data resampling with an affinity matrix and data imputation using the lightweight Swin Transformer (Swin-T) block. In the resampling part, the affinity matrix (as detailed in Sec. III) enables the adaptive selection and fusion of asynchronous, heterogeneous inputs. After resampling the fast data streams, the remaining data is fused with the slow stream and passed to the Swin-T block for further processing.

Fig. 8 illustrates the Swin-T block, as shown in Fig. 8, which consists of four stages. Initially, the fused features are segmented into equal patches by a patch-splitting module, allowing the model to learn the characteristics of missing data locally, particularly suited to data imputation tasks. To better handle multi-modal inputs, we redefine the traditional $Height \times Width$ or $Height \times Width \times Depth$ configurations.

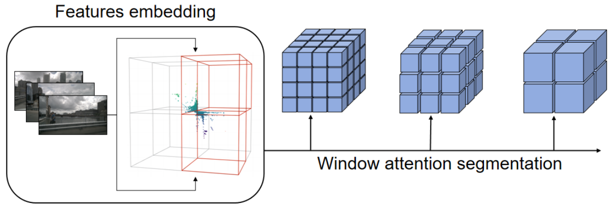


Fig. 6. Shifted Window applied in RGB images and cloud point.

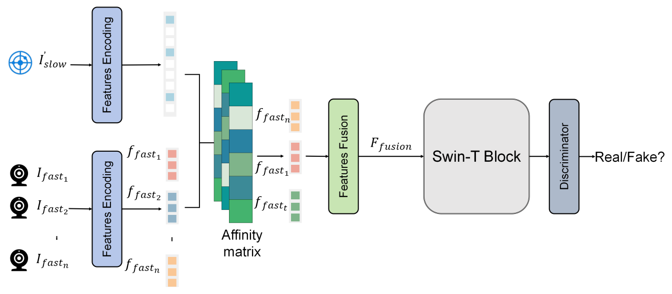


Fig. 7. Illustration of multi-modal lightweight Swin Transformer (MMLST) for real-time data imputation on resource-limited mobile devices.

Instead, we specify the number of patches, $N$, based on token count rather than shape, improving the model's adaptability to diverse input modalities. To enhance the Swin-T block's ability to learn inter-modal relationships, we further introduce a linear embedding layer that projects features into a higher-dimensional space, denoted by $C$. Finally, Swin Transformer blocks are applied to these patch tokens, maintaining the token count $N$ throughout Stage 1, which also includes the linear embedding.

As the network depth increases, the number of tokens decreases due to patch merging layers. The initial patch merging layer aggregates features from four neighboring patches. Subsequent Swin-T blocks transform features while maintaining a resolution of $N/4$ and doubling the output dimension to $2C$. This process, consisting of the patch merging and feature transformation, forms Stage 2. Stages 3 and 4 follow the same procedure, reducing the output resolutions to $N/16$ and $N/64$, respectively. Compared to the unoptimized Swin Transformer, we accelerate the patch merging process and reduce the number of stages from *many* to just *four*. This optimization significantly lowers the number of parameters, making the model more efficient for resource-constrained environments. Additionally, we maintain the basic setup of the two successive Swin Transformer blocks.

Notably, we *observe* that the shifted window mechanism of the Swin Transformer can seamlessly integrate with the proposed *affinity matrix*. Specifically, after the patch partitioning, the features from fast data and the corresponding features from slow data that require imputation are localized within specific patches. This strategic localization enhances the model's ability to effectively capture correlations between modalities, thereby improving the accuracy and efficiency of the feature fusion process in resource-constrained environments.

**Training of MMLST**. The training loss function consists of two components: the Swin-T loss $\mathcal{L}_{Swin}$, and a loss function for slow data stream characteristics $\mathcal{L}_*$, which measures the

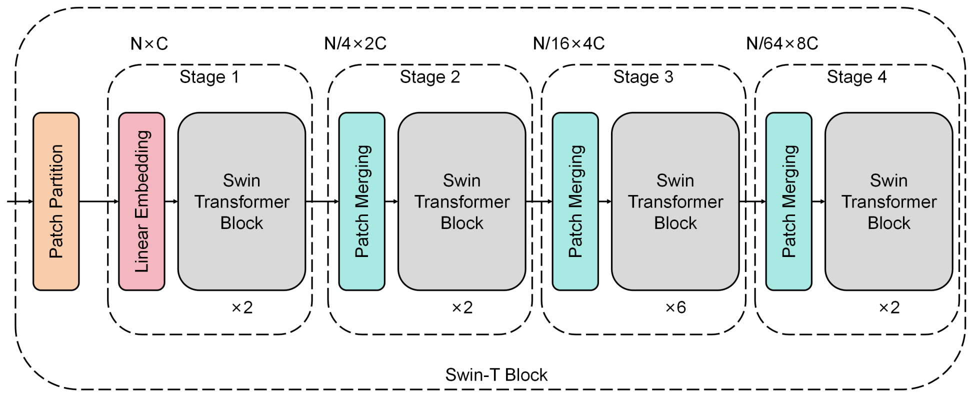


Fig. 8. Illustration of the Swin-T block in the proposed MMLST.

discrepancy between modalities. Inputs to the Swin-T block $S$ include complete fast data stream $\{I_{fast_1}, I_{fast_2}, ..., I_{fast_n}\}$ and the incomplete slow data stream $I'_{slow}$. This setup enables $S$ to generate the synthetic slow data $I^*_{slow}$. The loss function is formulated as follows:

$$\begin{aligned}\{F_{fast_1}, F_{fast_2}, ..., F_{fast_n}\} = \\ f_{fast}(\{I_{fast_1}, I_{fast_2}, ..., I_{fastn}\})\end{aligned} \tag{1}$$

$$F_{slow} = f_{slowR}(I'_{slow}) \tag{2}$$

$$F_{fusion} = g(h(\{F_{fast_1}, ..., F_{fast_n}\}), F_{slow}) \tag{3}$$

$$I^*_{slow} = S(F_{fusion}) \tag{4}$$

$$\begin{aligned}\mathcal{L}_{Swin} = \mathbb{E}_{I_{slow}}[logD(I_{slow})] \\ + \mathbb{E}_{I^*_{slow}}[log(1 - D(I^*_{slow}))]\end{aligned} \tag{5}$$

where the $S$ seeks to minimize Equ. 10, while the discriminator $D$ strives to maximize it.

For example, in the task of 3D object detection for autonomous driving assistance, the camera data is always the fast stream, whereas the LiDAR data is slow. Due to the characteristic nature of point cloud produced by LiDAR, we opt for the Chamfer distance loss $\mathcal{L}_{CD}$(Chamfer distance loss) between real point cloud $X$ and synthetic point cloud $Y$ as $\mathcal{L}_*$ which is expressed as follows:

$$X = \{x_i\}_{i=1}^{N} Y = \{y_i\}_{i=1}^{M}, X = I^*_{LiDAR} Y = I_{LiDAR} \tag{6}$$

$$\mathcal{L}_{CD} = \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 + \sum_{y \in Y} \min_{x \in X} \|y - x\|_2^2 \tag{7}$$

For each point $x$ in point cloud $X$, we identify the nearest point $y$ in point cloud $Y$ and compute the squared Euclidean distance between them, summing these values for all points in $X$. Similarly, for each point $y$ in $Y$, we find the nearest point $x$ in $X$, calculate the squared Euclidean distance between them, and sum these values for all points in $Y$. Thus, our final objective function becomes:

$$\mathcal{L} = arg \min_{S} \max_{D} \mathcal{L}_{Swin} + \lambda_{CD}\mathcal{L}_{CD} \tag{8}$$

where $\lambda_{CD}$ is a hyper-parameter to tune the data fitting and model complexity.

## B. *Affinity-aware Sub-graph Selection*

Existing data imputation methods, such as those discussed in [34], [35], typically presuppose a fixed set of modalities for data fusion and imputation, assuming static consistency and complementarity based on predetermined sensing patterns. However, they often fail in dynamic mobile environments, which are also characterized by resource constraints [17]. Studies [36], [37] reveal potential drawbacks. Specifically, [36] shows that adding more modalities without *key* ones can introduce noise and redundancy, degrading performance. Similarly, [37] reports that not all modalities contribute positively, with some introducing unnecessary noise. To address this, AdaFlowLite applies *affinity-aware sub-graph selection* to selectively choose modalities that enhance fusion for better imputation (as we will show in Sec. V-F1).

For instance, in the real-world deployment of AdaFlowLite in autonomous driving assistance systems, which integrates two LiDAR sensors and five cameras, we demonstrate the effectiveness of our approach under practical, resource-constrained conditions. The LiDARs are strategically positioned at the top and front, while the cameras capture images from various perspectives, including front, front-left, left, front-right, and right. These sensors correspond to sensors A∼G in Fig. 2a and Fig. 2b. Specifically, for the top LiDAR, which produces 360° point clouds in the slower data stream, we set a missing rate of 75% to mimic the delay typically observed in slow data streams, as shown in Fig. 2a. Given the broad sensing view of sensor A, its criteria for selecting fusion modalities should prioritize *complementarity* over *consistency*. Thus, fast data in sensor A with smaller cosine similarity are preferred for fusion to match the slow data stream, as detailed in Sec. 2a. Here, smaller cosine similarity, indicating larger average cosine distances, denotes a high level of *complementarity* between heterogeneous modalities. With a 75% missing rate of front LiDAR sensor B, selecting RGB images from other sensors with higher cosine similarity provides limited benefits in fusion accuracy. Similarly, for point clouds generated by other LiDAR sensors in the slow stream, we also maintain a 75% missing rate.

We note that the imputation may *vary* according to different data arrival patterns. *Affinity-aware sub-graph selection* can be described as follow: given a set of heterogeneous modalities

$M = \{m_1, m_2, \ldots, m_N\}$ as input, output the map-wise complementarity or consistency decision $m_{fi} \rightarrow m_{sj}$, where $m_{sj}$ is the slow modality and $m_{fi}$ is the fast modality. It can be formulated as $m_{fi} \rightarrow m_{sj} = \mathrm{F}(M)$. For $\mathrm{F}(M)$, we assess complementarity and consistency using the intersection over union between modalities. The intersection over union between modalities is defined as:

$$v_{ij} = \frac{\mathrm{FOV}_{m_{sj}} \cap \mathrm{FOV}_{m_{fi}}}{\mathrm{FOV}_{m_{sj}} \cup \mathrm{FOV}_{m_{fi}}} \tag{9}$$

And if $v_{ij} \neq 0$, $v_{ij} \in V_j$. For each $m_{sj} \in S$, we set:

$$m_{fi} \rightarrow m_{sj} = \begin{cases} Consistency\ , & \exists v_{ij} = 0 \\ Complementarity, & \nexists\, v_{ij} = 0 \end{cases} \tag{10}$$

For $m_{sj}$, its' criteria of affinity has settle down, $a_{ij}$ is the affinity between $m_{fi}$ and $m_{sj}$, Then we have:

$$A = \sum^{j} \sum_{i=1}^{k} a_{ij} \tag{11}$$

Where the value of $k$ as shown below:

$$k = \lfloor |V_j| \times r \rfloor \tag{12}$$

Where $r$ is the data missing rate of partial (slow) data and $|V_j|$ is the number of intersecting FOV sensors. We need to discuss for value $k$:

$$k = \begin{cases} k, k \neq 0 \\ 1, k = 0 \ |V_j| \neq 0 \\ 0, k = 0 \ |V_j| = 0 \end{cases} \tag{13}$$

By maximizing $A$ (Equ. 16), we obtain the $m_{fi} \rightarrow m_{sj}$ map.

Based on the arrival speed and availability of input data streams, AdaFlowLite adaptively determines the selection of *affinity sub-graph* for data fusion by maximizing $A$, *i.e.* which modality and modality $S$ to fuse, and which are the target tasks $F$ using the selected affinity sub-graph, to maximize the imputation effect. In particular, based on the affinity matrix as discussed in Sec. III, the sub-graph selection criteria for runtime fusion are dynamically determined by the specific requirements of the downstream task, influenced by the asynchrony and heterogeneity of input data.

## V. Experiments

### A. Setups

**Implementation.** AdaFlowLite is implemented using Python 3.9 and PyTorch 1.11 on a server equipped with an RTX3090 GPU, Intel(R) Xeon(R) Gold 6133 CPU @ 2.50GHz, and 256GB RAM. For distributed multi-modal data on mobile devices, we realize both simulation with real-world datasets and actual collections. We also simulated the bandwidth of the real car Internet, which is 100Mbps. In the implementation of the affinity matrix (Sec. III ), we designate LiDAR as the primary modality and cameras as secondary modalities. For the MMLST architecture (Sec. IV-A), we employ two feature-capturing networks to process the raw data. These networks, via the affinity matrix, select the appropriate modalities for fusion. The resultant fusion features are then inputted into the Swin-T block to synthesize the primary modality data.

**Tasks, Datasets, and Model.** We experiment with 4 real-world distributed multi-modal applications. They are widely used to evaluate multi-modal fusion performance [12], [38], [39].

- **BEVFusion** ($T_1$) is a real-world multi-modal 3D object recognition task using a bird's-eye view. The **dataset** (nuScenes [12]) contains 2000 training frames and 81 testing frames, each featuring 6 camera views (fast data) and 1 LiDAR views (slow data). The **model** for BEVFusion extracts a bird's-eye view (BEV) from both the camera and LiDAR branches.
- **PointPillars** ($T_2$) is another real-world 3D object recognition model. The **dataset** from KITTI [38] includes 2500 frames for training, and 81 frames for testing, all collected from actual autonomous driving systems, with each frame featuring 4 camera views (fast data) and 1 LiDAR view (slow data) The **model** projects both camera and LiDAR data into a unified space and merges multiple views.
- **LLM-based driver behavior recognition** ($T_3$) employs **Large Language Model (LLM)** [40] to categorize driver behaviors into 12 classes using the Drive&Act dataset [39], which comprises 6 infrared camera views (fast data) and 1 standard camera view (slow data).
- **LLM-based event recognition** ($T_4$) employes **LLM** for audio-visual event recognition across 28 categories using the AVE dataset [41] which contains audio (fast data) and video (slow data).

**Baselines.** We adopt five a-/synchronous fusion methods as comparison baselines to validate the affinity matrix.

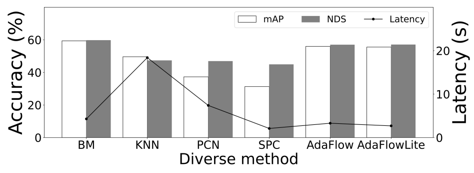
- **Synchronous fusion** (blocking mechanism, BM) [24]: the inference pipeline waits until all input data arrive. It provides an upper-bound accuracy.
- **Asynchronous fusion** (non-blocking mechanism):
  - Sparse Point Cloud (SPC) [25]: discards the slow data for inference which is the fastest method but exhibits the lowest accuracy.
  - PCN [26]: is a DL model to impute 3D point clouds. When the point cloud is missing due to data asynchrony, we use it to complete the point cloud.
  - KNN [13]: is a traditional imputation method, which classifies an input by majority class among its $K$ nearest neighbors to selectively discard data.
  - AdaFlow [19]: has the same data resampling effect as AdaFlowLite under fixed modalities. We use it to test the Swin-T block's lightweight characteristics.
  - Only Swin-T block: has the same imputation network structure as AdaFlowLite, but selects fixed or random fast data to impute the slow data. We use it to test the effectiveness of the affinity matrix.

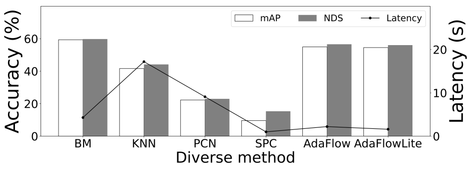
### B. Performance Comparison

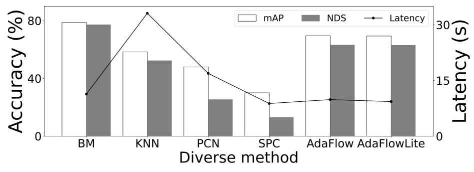
We evaluate the inference accuracy and latency of AdaFlowLite against five baseline methods (BM, SPC, PCN, KNN and AdaFlow) across BEVFusion ($T_1$) and PointPillars ($T_2$) tasks under various missing rates of slow data, which simulate data asynchrony. For example, a 25% missing rate of slow data represents a delay of 26ms in dataset nuScenses with 100Mbps.

Fig. 9 shows the results. *First*, AdaFlowLite exhibits the best trade-off between inference accuracy and latency compared to the three asynchronous fusion methods (SPC, PCN, and KNN), and achieves almost as good accuracy as AdaFlow. *Second*, AdaFlowLite outperforms the three asynchronous fusion methods (SPC, PCN, and KNN) under various missing rates of slow data. For example, as shown in Fig. 9b, with 75% missing rate in BEV, AdaFlowLite demonstrates accuracy improvements of 44.9%, 32.2%, 12.9% compared to SPC, PCN, and KNN, respectively. *Third*, with acceptable accuracy, AdaFlowLite achieves the lowest latency compared to other baselines. As shown in Fig. 9d, with 50% missing rate of slow LiDAR data in PointPillars, AdaFlowLite achieves 49.0%, 80.4%, 73.1%, 8.3% latency reduction compared to BM, KNN, PCN, and AdaFlow. Besides, AdaFlowLite improves 62.1% accuracy compared to SPC with only 0.5s slower than SPC for 81 frames.

**Summary.** AdaFlowLite offers the optimal balance between accuracy and latency, making it a promising solution for latency-sensitive tasks that often involve asychronous data.


(a) BEVFusion, missing 50% LiDAR.


(b) BEVFusion, missing 75% LiDAR.


(c) PointPillars, missing 25% LiDAR.

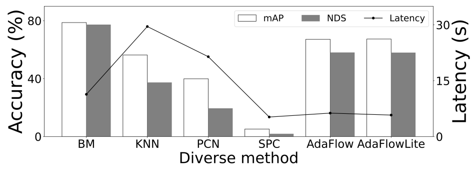

(d) PointPillars, missing 50% LiDAR.

Fig. 9. Comparison of accuracy and inference latency between AdaFlowLite and baselines in two tasks, under varying missing rates of slow data (*i.e.* lagging delay).

## C. Parameter Comparison

We evaluate the numbers of parameters, FLOPs, and storage requirements of AdaFlowLite against AdaFlow. Tab. I shows the results. AdaFlowLite achieves a reduction of 49.5%, 51.5%, and 49.5% in the number of parameters, FLOPs, and storage requirements, respectively, compared to AdaFlow. As discussed in Sec. V-B, AdaFlowLite maintains nearly identical accuracy with a 50% reduction in these metrics, making it more suitable for resource-constrained environments.

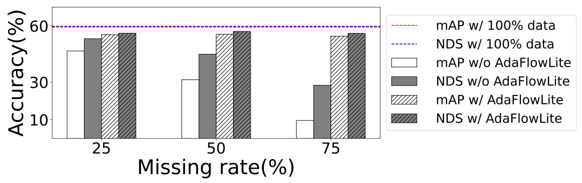


Fig. 10. AdaFlowLite's imputation performance in BEVFusion under different missing rates of slow data.

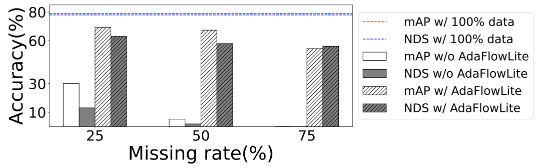


Fig. 11. AdaFlowLite's imputation performance in PointPillars under different missing rates of slow data.

TABLE I
COMPARISON OF PARAMETERS BETWEEN ADAFLOW AND AdaFlowLite.

| Models | Parameters | FLOPs(GFLOPs) | Storage Requirements(MB) |
|---|---|---|---|
| AdaFlow | 354233600 | 4.648 | 1351.29 |
| AdaFlowLite | 178836346 | 2.256 | 682.21 |

## D. AdaFlowLite's *Data Imputation Performance*

We evaluate AdaFlowLite's data imputation module on BEVFusion ($T_1$) and PointPillars ($T_2$) under different data missing rates of slow data. *First*, AdaFlowLite efficiently prevents significant accuracy decline at varying slow data missing rates. As shown in Fig. 10, AdaFlowLite suffer only 4.1%, 3.8%, and 4.8% accuracy decline with data missing rates at 25%, 50%, and 75%. While SPC incurs up to 12.7%, 28.1% and 49.8% accuracy drop, respectively. *Second*, AdaFlowLite is more necessary when suffering from *extreme* slow data missing. As shown in Fig. 11, AdaFlowLite improves 54.3% mAP and 55.9% NDS compared to without data imputation under data missing rates at 75%. Comparatively, without AdaFlowLite's data imputation module, a 75% data missing rate causes an 78.6% decline in accuracy and a 77.2% drop in NDS, making the system unusable.

**Summary.** Under various asynchrony levels (*i.e.* slow data missing rates), AdaFlowLite maintains system effectiveness by preventing accuracy declines. Particularly in extreme asychronous cases, AdaFlowLite avoids drastic accuracy drop.

## E. Generalizing to Diverse Input Data

We evaluate AdaFlowLite's generalization ability on three diverse tasks, *i.e.* BEVFusion ($T_1$), and two LLM-based tasks ($T_3, T_4$), across different data missing rates of slow data. As shown in Fig. 12. *First*, AdaFlowLite can adapt to various multi-modal tasks. In detail, on $T_1$, $T_3$, and $T_4$, AdaFlowLite

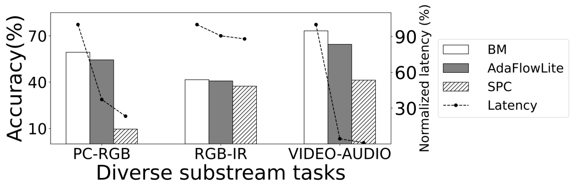


Fig. 12. Performance over diverse downstream tasks.

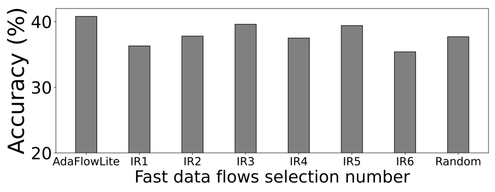


Fig. 13. Comparison between AdaFlowLite and Swin-T block.

achieve 44.8%, 3.4%, and 23.2% accuracy improvements compared to SPC, a typical asynchronous baseline. *Second*, AdaFlowLite performs best when the data volume ratio between fast and slow modalities is large. As shown in Fig. 14. For data ratios of 1:4, 8:9, and 1:70, AdaFlowLite reduces latency by 62.8%, 9.4%, and 95.6%, respectively, compared to BM, a synchronous method with upper-bound accuracy. These gains are attributed to the one-fit-all MMLST for dynamic data imputation and leverage a class attention mechanism for maximum data imputation.

### *F. Performance of Affinity Matrix*

*1) Validity Verification:* We test AdaFlowLite's affinity matrix on LLM task ($T_3$) under 10% data missing rates of slow data. *First*, AdaFlowLite achieves higher accuracy than Swin-T block's 6 fixed selections, as shown in Fig. 13. AdaFlowLite's accuracy is 40.8% and 6 Swin-T block's 6 fixed selections is 36.3%, 37.8%, 39.6%, 37.5%, 39.4% and 35.4%, respectively. The inference derived by AdaFlowLite's generalized affinity matrix is also higher than Swin-T block's random selection (37.7%). *Second*, AdaFlowLite achieves higher accuracy with just one fast modality input, as shown in Fig. 14, while accuracy drops to 29.9% when all modalities are used together. This proves that the affinity matrix can always select the most suitable combination of available (fast) data for missing (slow) data imputation in diverse situations and supports the argument in Sec. IV-B.

*2) Comparison to Monte Carlo:* We compare the affinity matrix outperformed by AdaFlowLite and Monte Carlo in Fig. 15 and Fig. 16. For each matrix, the x-axis represents different camera views, the y-axis represents different LiDAR views. And elements in the matrix are the affinity value between a LiDAR view and a camera view. In particular, we use the Monte Carlo [42] method here to establish relationships between modalities, which can be regarded as the baseline (*i.e.* ground truth). We chose the Monte Carlo method for its efficiency in approximating complex problems through repeated random sampling, achieving reliable solutions quickly.

To simulate mobile scenarios where sensors need to be added, we mask the rear Lidar and front camera in the Waymo dataset in the first frame, and add the rear Lidar and front camera in the 10-th frame and the 20-th frame, respectively.

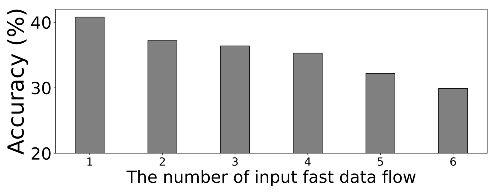
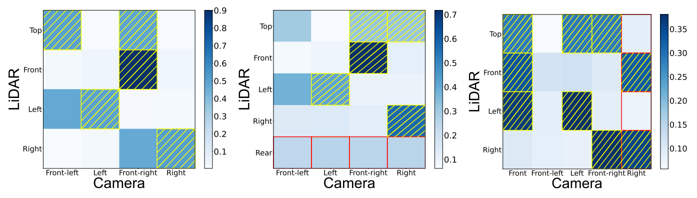


Fig. 14. Performance on diverse modality numbers.



(a) 1st frame (b) 10th frame (c) 20th frame

Fig. 15. Visualization of AdaFlowLite's affinity matrix.

*First*, AdaFlowLite's affinity matrix accurately captures the relationships between modalities as shown in Fig. 16a and Fig. 15a. And AdaFlowLite's affinity matrix can dynamically adapt to changes in inter-modal relationships, as shown in Fig. 15, the same modal settings exhibited different inter-modal relationships at different times. These features confirm that AdaFlowLite's affinity matrix matches the functionality of AdaFlow in handling dynamic modal environments. *Second*, AdaFlowLite's affinity matrix demonstrates exceptional scalability by dynamically accommodating new sensors. As shown in Fig. 15 and Fig. 16, the matrix initially comprises four rows and four columns. Integrating the slow sensor rear Lidar by the 10th frame expands the matrix to five rows and four columns. Subsequently, with the addition of the fast sensor front camera by the 20th frame, the matrix adjusts again to four rows and five columns. This ability to adapt its structure with introducing new sensors highlights the robustness and flexibility of AdaFlowLite in diverse mobile scenarios.

### *G. Case Study*

Fig. 17 illustrates the latency breakdown in conventional end-to-end multi-modal object detection systems for autonomous driving, categorized into data pre-processing (*e.g.* handling or imputing slow data) and executing inference. AdaFlowLite enhances system performance by utilizing affinity-aware data imputation for pre-processing, thereby reducing the waiting time associated with slow data and improving overall detection responsiveness. To evaluate AdaFlowLite, we generated a 10,000-frame autonomous driving dataset using the Carla[43] autopilot simulator. As demonstrated in Fig. 18, the system processes inputs from six camera views (fast data) and one LiDAR view (slow data). We benchmark AdaFlowLite against three asynchronous methods, *i.e.* AdaFlow, KNN, and PCN, comparing data pre-processing latency and imputation quality. We test data quality using Maximum Mean Discrepancy (MMD) and Chamfer Distance (CD) error metrics, comparing the imputed data with actual sensor data.

Fig. 19c shows the results. AdaFlowLite reduces the overall latency of the pre-processing phase from 0.346s, 0.246s,

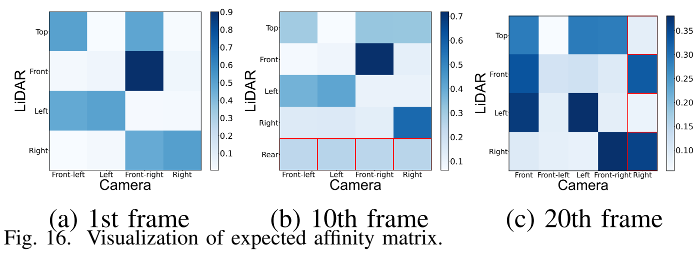


Fig. 16. Visualization of expected affinity matrix.

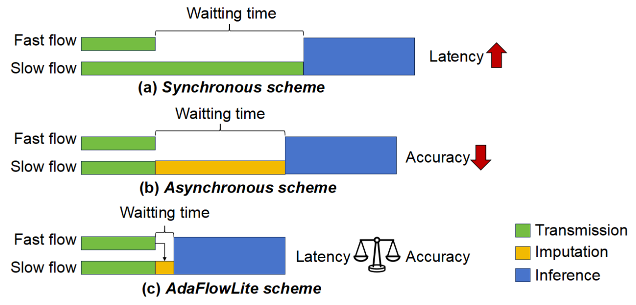


Fig. 17. Paradigms of AdaFlowLite and baselines.

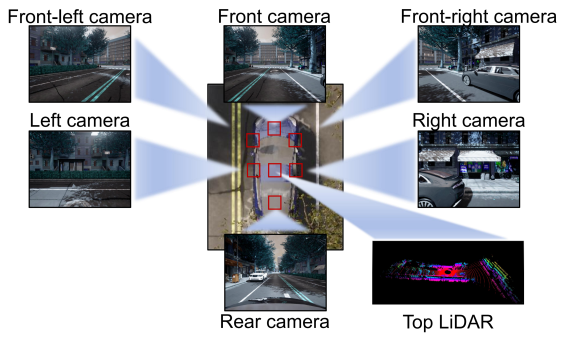


Fig. 18. Illustration of seven sensors (six cameras and one LiDAR) on vehicle.

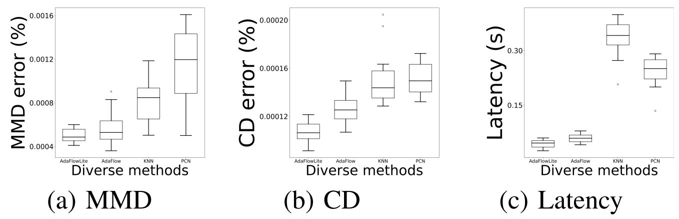


Fig. 19. Comparison between AdaFlowLite and asynchronous baselines (AdaFlow, KNN, PCN) in the self-collect dataset under 66% data missing rates of LiDAR.

0.141s, 0.059s to 0.04s, compared to KNN, PCN, BM ,and AdaFlow respectively. While when imputing data similar to those collected by real sensors (see Fig. 19a and Fig. 19b), AdaFlowLite achieves latency reductions of $1.5 \sim 8.7\times$ than BM, PCN, KNN, and AdaFlow. In practical urban settings, where vehicles commonly travel at 36 km/h, with such latency reduction, AdaFlowLite reduces the detection distance interval from 0.59m-3.46m to just 0.4m. This significant improvement enhances the timely detection of pedestrians and vehicles, helping accident prevention.

We deploy AdaFlowLite with three different bandwidths, *i.e.* 50bps, 75Mbps, and 100Mbps on NVIDIA AGX Xaxier, similar Mercedes in-car chips. As shown in Fig. 20, AdaFlowLite remains efficient even under low bandwidth. Compared to BM, it reduces latency by 49% at 50Mbps while keeping CD error within acceptable limits (*i.e.* $\leq 0.00015$). We also test AdaFlowLite's energy cost on the AGX Xavier. As shown in Fig. 21. AdaFlowLite consumes approximately 15W, which represents a 31% reduction in energy consumption compared to AdaFlow. This enhanced efficiency makes AdaFlowLite more suitable for resource-constrained environments.

## VI. Related Work

**Multi-modal Inference in Mobile Applications.** Multi-modal inference has demonstrated success across a range of mobile sensing tasks such as classification [15], [16], [44], [18], [45], object detection [46], [47], [48], [49], [50], [51], [52], [53], segmentation [54], [55], [56], speech recognition [57], autonomous driving [12], [38], [58], [59], and medical image segmentation [60]. For instance, the nuScenes dataset [12] illustrates how integrating multi-modal data can enhance perception in autonomous driving, improving the effectiveness and user experience in driving and traffic scenarios.

**A-/Synchronous Multi-modal Inference.** Multi-modal inference processes data from multiple *asynchronous* flows, where the slowest flow often increases waiting time. Existing methods are divided into *synchronous blocking* and *asynchronous non-blocking* approaches. Synchronous methods, like frame sampling [61], wait for all data to arrive, optimizing latency by transmitting keyframes, but risking accuracy if slow data is low quality. Asynchronous methods, such as Tianxing *et al.* [17], use predictive completion to reduce wait time but are limited by fixed input modality configurations. AdaFlowLite leverages cross-modal affinity for attention-based fusion, enabling efficient asynchronous inference across diverse input modalities, regardless of number or type. Based on [19], AdaFlowLite utilizes MMLST to further optimize the number of parameters so that the framework can run in resource-constrained environment.

**Cross-modal Affinity Metrics in Multi-modal Inference.** The affinity metric identifies which modalities can be omitted without significantly impacting cross-modal inference accuracy. Existing approaches to quantify affinity fall into two categories: *model-based* and *function-based*. Model-based methods, like ReID [27], integrate the affinity matrix into the loss function during training, capturing affinity between specific modalities but requiring retraining for each new input. Function-based methods, such as Taskonomy [20], compute affinity from statistical data features without modifying the model. AdaFlowLite leverages the flexible function-based approach of the preliminary AdaFlow [19], and extends its capabilities with a prior-free affinity refinement module for seamlessly integrating new sensors into the structured modality affinity matrix by utilizing *prior-free analysis* specifically designed to effectively manage *unseen* modalities. In contrast to existing methods, which are restricted to predefined modalities and lack adaptability in accommodating new sensor types or quantities, AdaFlowLite introduces a modality-agnostic and scalable affinity quantification method. This approach significantly enhances cross-modal inference, particularly in dynamic

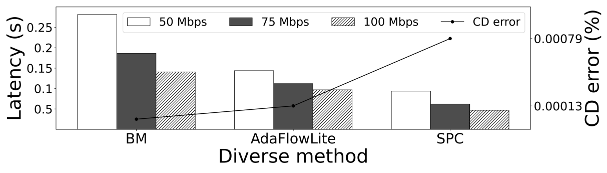


Fig. 20. Performance with diverse bandwidths.

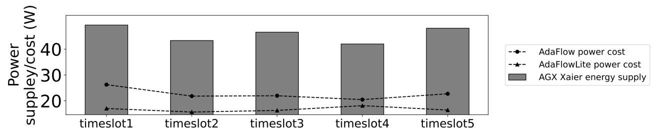


Fig. 21. Energy cost of AdaFlowLite on AGX Xavier.

and asynchronous settings, addressing critical challenges in the fusion and analysis of sensor data.

## VII. Conclusion

To tackle the challenges of data *asynchrony* and *heterogeneity* in distributed mobile environments, this paper presents AdaFlowLite, a system designed to perform inference as soon as partial asynchronous data becomes available, thereby enhancing operational efficiency. *First*, AdaFlowLite introduces a generalized modality-scalable affinity matrix to model and optimize inference processes, enabling dynamic adaptation across mobile scenarios of inputs and facilitating effective multi-modal fusion. Additionally, AdaFlowLite employs a prior-free affinity refinement module for seamlessly integrating *unseen* modalities. *Second*, AdaFlowLite employs a one-fit-all multi-modal lightweight Swin Transformer (MMLST) that is specifically designed for high-accuracy and low-latency inference. This model is not only lightweight, enabling minimal computational overhead, but also non-blocking, which facilitates continuous processing across various input modalities without the need for retraining. This lightweight architecture is crucial for maintaining performance efficiency in resource-constrained mobile environments. Through evaluations on real-world multi-modal tasks, AdaFlowLite has demonstrated significant reductions in inference latency and substantial improvements in accuracy, all while maintaining a low resource footprint. Looking forward, there is potential to expand AdaFlowLite's capabilities to include adaptive affinity-based data imputation across a swarm of IoT devices.

## Acknowledgment

This work was partially supported by the National Science Fund for Distinguished Young Scholars (62025205) and the National Natural Science Foundation of China (No. 62032020, 62102317,62472354).

## References

[1] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018.

[2] Sicong Liu, Bin Guo, Cheng Fang, Ziqi Wang, Shiyan Luo, Zimu Zhou, and Zhiwen Yu. Enabling resource-efficient aiot system with cross-level optimization: A survey. *IEEE Communications Surveys & Tutorials*, 2023.

[3] Google automotive service. https://developers.google.com/cars?hl=zh-cn.

[4] Tesla autopilo. https://www.tesla.com/autopilot.

[5] Microsoft azure kinect dk. https://learn.microsoft.com/en-us/azure/kinect-dk.

[6] Apple healthkit. https://developer.apple.com/health-fitness.

[7] Meta oculus rift. https://www.facebook.com/marketplace/category/oculus-rifts/.

[8] Abhinav Valada, Rohit Mohan, and Wolfram Burgard. Self-supervised model adaptation for multimodal semantic segmentation. *International Journal of Computer Vision*, 128(5):1239–1285, 2020.

[9] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. *ACM SIGARCH Computer Architecture News*, 45(1):615–629, 2017.

[10] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani. Deep learning for iot big data and streaming analytics: A survey. *IEEE Communications Surveys & Tutorials*, 20(4):2923–2960, 2018.

[11] Ziqi WANG Sicong LIU, Bin GUO et al. Deepswarm: towards swarm deep learning with bi-directional optimization of data acquisition and processing. *Frontiers of Computer Science*, 19(3):193501, 2025.

[12] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of CVPR*, pages 11621–11631, 2020.

[13] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[14] Xiaomin Ouyang, Xian Shuai, Jiayu Zhou, Ivy Wang Shi, Zhiyuan Xie, Guoliang Xing, and Jianwei Huang. Cosmo: contrastive fusion learning with small data for multimodal human activity recognition. In *Proceedings of MobiCom*, pages 324–337, 2022.

[15] Danfeng Hong, Jingliang Hu, Jing Yao, Jocelyn Chanussot, and Xiao Xiang Zhu. Multimodal remote sensing benchmark datasets for land cover classification with a shared and specific feature learning model. *ISPRS Journal of Photogrammetry and Remote Sensing*, 178:68–80, 2021.

[16] Ajian Liu, Jun Wan, Sergio Escalera, Hugo Jair Escalante, Zichang Tan, Qi Yuan, Kai Wang, Chi Lin, Guodong Guo, Isabelle Guyon, et al. Multi-modal face anti-spoofing attack detection challenge at cvpr2019. In *Proceedings of CVPR Workshop*, pages 0–0, 2019.

[17] Tianxing Li, Jin Huang, Erik Risinger, and Deepak Ganesan. Low-latency speculative inference on distributed multi-modal data streams. In *Proceedings of MobiSys*, pages 67–80, 2021.

[18] Xiaochen Li, Sicong Liu, Zimu Zhou, Bin Guo, Yuan Xu, and Zhiwen Yu. Echopfl: Asynchronous personalized federated learning on mobile devices with on-demand staleness control. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 8(1):1–22, 2024.

[19] Fenmin Wu, Sicong Liu, Kehao Zhu, Xiaochen Li, Bin Guo, Zhiwen Yu, Hongkai Wen, Xiangrui Xu, Lehao Wang, and Xiangyu Liu. Adaflow: Opportunistic inference on asynchronous mobile data with generalized affinity control, 2024.

[20] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of CVPR*, pages 3712–3722, 2018.

[21] Qingyang Zhang, Haitao Wu, Changqing Zhang, Qinghua Hu, Huazhu Fu, Joey Tianyi Zhou, and Xi Peng. Provable dynamic fusion for low-quality multimodal data. In *International conference on machine learning*, pages 41753–41769. PMLR, 2023.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] Tijmen Tieleman and Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *COURSERA Neural Networks Mach. Learn*, 17, 2012.

[24] Juexing Wang, Guangjing Wang, Xiao Zhang, Li Liu, Huacheng Zeng, Li Xiao, Zhichao Cao, Lin Gu, and Tianxing Li. Patch: A plug-in framework of non-blocking inference for distributed multimodal system. *Proceedings of UbiComp*, 7(3):1–24, 2023.

[25] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *Proceedings of ICML*, pages 689–696, 2011.

[26] Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*, pages 728–737. IEEE, 2018.

[27] Yan Lu, Yue Wu, Bin Liu, Tianzhu Zhang, Baopu Li, Qi Chu, and Nenghai Yu. Cross-modality person re-identification with shared-specific feature transfer. In *Proceedings of CVPR*, pages 13379–13389, 2020.

[28] Samuel Ubellacker, Aaron Ray, James M Bern, Jared Strader, and Luca Carlone. High-speed aerial grasping using a soft drone with onboard perception. *npj Robotics*, 2(1):5, 2024.

[29] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[30] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[31] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[32] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[34] Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning*, pages 5689–5698. PMLR, 2018.

[35] Pierre-Alexandre Mattei and Jes Frellsen. Miwae: Deep generative modelling and imputation of incomplete data sets. In *International conference on machine learning*, pages 4413–4423. PMLR, 2019.

[36] Qi Wang, Liang Zhan, Paul Thompson, and Jiayu Zhou. Multimodal learning with incomplete modalities by knowledge distillation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1828–1838, 2020.

[37] Xin Luna Dong, Barna Saha, and Divesh Srivastava. Less is more: Selecting sources wisely for integration. *Proceedings of the VLDB Endowment*, 6(2):37–48, 2012.

[38] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

[39] Manuel Martin, Alina Roitberg, Monica Haurilet, Matthias Horne, Simon Reiß, Michael Voit, and Rainer Stiefelhagen. Drive&act: A multi-modal dataset for fine-grained driver behavior recognition in autonomous vehicles. In *Proceedings of ICCV*, pages 2801–2810, 2019.

[40] Jiaming Han, Kaixiong Gong, Yiyuan Zhang, Jiaqi Wang, Kaipeng Zhang, Dahua Lin, Yu Qiao, Peng Gao, and Xiangyu Yue. Onellm: One framework to align all modalities with language. In *Proceedings of CVPR*, pages 26584–26595, 2024.

[41] Yapeng Tian, Jing Shi, Bochen Li, Zhiyao Duan, and Chenliang Xu. Audio-visual event localization in unconstrained videos. In *Proceedings of ECCV*, pages 247–263, 2018.

[42] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of AAAI*, volume 30, 2016.

[43] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[44] Haiman Tian, Yudong Tao, Samira Pouyanfar, Shu-Ching Chen, and Mei-Ling Shyu. Multimodal deep representation learning for video classification. *World Wide Web*, 22:1325–1341, 2019.

[45] Xiaochen Li, Sicong Liu, Zimu Zhou, Yuan Xu, Bin Guo, and Zhiwen Yu. Classter: Mobile shift-robust personalized federated learning via class-wise clustering. *IEEE Transactions on Mobile Computing*, 2024.

[46] Wen-Da Jin, Jun Xu, Qi Han, Yi Zhang, and Ming-Ming Cheng. Cdnet: Complementary depth network for rgb-d salient object detection. *IEEE Transactions on Image Processing*, 30:3376–3390, 2021.

[47] Peng Sun, Wenhu Zhang, Huanyu Wang, Songyuan Li, and Xi Li. Deep rgb-d saliency detection with depth-sensitive attention and automatic multi-modal fusion. In *Proceedings of CVPR*, pages 1407–1417, 2021.

[48] Tao Zhou, Deng-Ping Fan, Ming-Ming Cheng, Jianbing Shen, and Ling Shao. Rgb-d salient object detection: A survey. *Computational Visual Media*, 7:37–69, 2021.

[49] Sicong Liu, Bin Guo, Ke Ma, Zhiwen Yu, and Junzhao Du. Adaspring: Context-adaptive and runtime-evolutionary deep model compression for mobile applications. *Proceedings of UbiComp*, 5(1):1–22, 2021.

[50] Lehao Wang, Zhiwen Yu, Haoyi Yu, Sicong Liu, Yaxiong Xie, Bin Guo, and Yunxin Liu. Adaevo: Edge-assisted continuous and timely dnn model evolution for mobile devices. *IEEE Transactions on Mobile Computing*, 2023.

[51] Sicong Liu, Yungang Wu, Bin Guo, Yuzhan Wang, Ke Ma, Liyao Xiang, Zhetao Li, and Zhiwen Yu. Caq: Toward context-aware and self-adaptive deep model computation for aiot applications. *IEEE Internet of Things Journal*, 9(21):20801–20814, 2022.

[52] Hongli Wang, Bin Guo, Jiaqi Liu, Sicong Liu, Yungang Wu, and Zhiwen Yu. Context-aware adaptive surgery: A fast and effective framework for adaptative model partition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(3):1–22, 2021.

[53] Sicong Liu, Hao Luo, XiaoChen Li, Yao Li, Bin Guo, Zhiwen Yu, YuZhan Wang, Ke Ma, YaSan Ding, and Yuan Yao. Adaknife: Flexible dnn offloading for inference acceleration on heterogeneous mobile devices. *IEEE Transactions on Mobile Computing*, 2024.

[54] Jinming Cao, Hanchao Leng, Dani Lischinski, Daniel Cohen-Or, Changhe Tu, and Yangyan Li. Shapeconv: Shape-aware convolutional layer for indoor rgb-d semantic segmentation. In *Proceedings of ICCV*, pages 7088–7097, 2021.

[55] Xinxin Hu, Kailun Yang, Lei Fei, and Kaiwei Wang. Acnet: Attention based network to exploit complementary features for rgbd semantic segmentation. In *2019 IEEE International conference on image processing (ICIP)*, pages 1440–1444. IEEE, 2019.

[56] Daniel Seichter, Mona Köhler, Benjamin Lewandowski, Tim Wengefeld, and Horst-Michael Gross. Efficient rgb-d semantic segmentation for indoor scene analysis. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 13525–13531. IEEE, 2021.

[57] Stavros Petridis, Themos Stafylakis, Pingehuan Ma, Feipeng Cai, Georgios Tzimiropoulos, and Maja Pantic. End-to-end audiovisual speech recognition. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 6548–6552. IEEE, 2018.

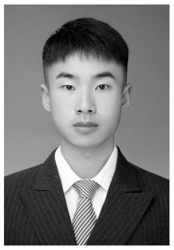
[58] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of CVPR*, pages 2446–2454, 2020.

[59] Cheng Fang, Sicong Liu, Zimu Zhou, Bin Guo, Jiaqi Tang, Ke Ma, and Zhiwen Yu. Adashadow: Responsive test-time model adaptation in non-stationary mobile environments. In *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems*, pages 295–308, 2024.

[60] Zhe Guo, Xiang Li, Heng Huang, Ning Guo, and Quanzheng Li. Deep learning-based image segmentation on multimodal medical imaging. *IEEE Transactions on Radiation and Plasma Medical Sciences*, 3(2):162–169, 2019.

[61] Tan Zhang, Aakanksha Chowdhery, Paramvir Bahl, Kyle Jamieson, and Suman Banerjee. The design and implementation of a wireless video surveillance system. In *Proceedings of MobiCom*, pages 426–438, 2015.

**Sicong Liu** received the PhD degree in school of computer science and technology from Xidian University in 2020. Now she is an associate professor with school of computer science, Northwestern Polytechnical University. Her research interests include mobile computing and resource-efficient mobile deep learning. She has served as the TPC member of MobiSys 2021 and BigCom 2021.



**Fengmin Wu** is a Doctoral student at Northwestern Polytechnical University, China. He received B.E. from China University of Petroleum (Beijing) in 2023. His research interests include resource-efficient mobile deep learning and mobile systems.

**Yuan Gao** is a Master's student at Northwestern Polytechnical University, China. He received his B.E. from Northwestern Polytechnical University in 2023. His interest is in Heterogeneous Multi-Agent Collaborative Sensing and Perception.

**Bin Guo** received the PhD degree in computer science from Keio University, Japan, and then was a postdoc researcher with Institut Telecom SudParis, France. He is a professor with Northwestern Polytechnical University, China. His research interests include ubiquitous computing, mobile crowd sensing, and HCI. He has served as an associate editor of the IEEE Communications Magazine and the IEEE Transactions on Human-Machine-Systems, the guest editor of the ACM Transactions on Intelligent Systems.

**Zimu Zhou** is currently an assistant professor in the Department of Data Science, City University of Hong Kong. He obtained his Ph.D. from the Hong Kong University of Science and Technology in 2016 and B.E. from Tsinghua University in 2011. His research interest lies broadly in mobile and ubiquitous computing, with a focus on AIoT. zimuzhou@cityu.edu.hk Zimu Zhou's research is supported by CityU APRC grant (No. 9610633).

**Hongkai Wen** is a professor (Chair in Machine Learning Systems) at the Department of Computer Science, the University of Warwick, where he lead the AI Systems Lab. He is a Fellow of the Alan Turing Institute, the UK's national institute for data science and AI. At the Turing, he is also an Independent Scientific Advisor for the BridgeAI programme, and a member of the Turing Research Ethics (TREx) team.

**Zhiwen Yu** received the Ph.D. degree in computer science from Northwestern Polytechnical University, Xi'an, China, in 2005. He is currently the vice president of Harbin Engineering University and a professor at the School of Computer Science, Northwestern Polytechnical University. He was an Alexander Von Humboldt Fellow with Mannheim University, Germany, and a Research Fellow with Kyoto University, Kyoto, Japan. His research interests include ubiquitous computing, HCI, and mobile sensing and computing.