

AdaFlow: Opportunistic Inference on Asynchronous Mobile Data with Generalized Affinity Control

Fengmin Wu
fenny@mail.nwp.edu.cn
Northwestern Polytechnical University

Xiaochen Li
lxcuwang365@163.com
Northwestern Polytechnical University

Hongkai Wen
hongkai.wen@warwick.ac.uk
University of Warwick

Sicong Liu*
scliu@mail.nwp.edu.cn
Northwestern Polytechnical University

Bin Guo
guob@mail.nwp.edu.cn
Northwestern Polytechnical University

Xiangrui Xu
xiangruixu@mail.nwp.edu.cn
Northwestern Polytechnical University

Xiangyu Liu
liuxy01@mail.nwp.edu.cn
Northwestern Polytechnical University

Kehao Zhu
zhukehao@mail.nwp.edu.cn
Northwestern Polytechnical University

Zhiwen Yu
zhiwenyu@mail.nwp.edu.cn
Northwestern Polytechnical University and Harbin Engineering University

Lehao Wang
lehaowang@mail.nwp.edu.cn
Northwestern Polytechnical University

Abstract

The rise of mobile devices equipped with numerous sensors, such as LiDAR and cameras, has spurred the adoption of multi-modal deep intelligence for distributed sensing tasks, such as smart cabins and driving assistance. However, the arrival times of mobile sensory data vary due to modality size and network dynamics, which can lead to delays (if waiting for slower 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. AdaFlow 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 affinity attention-based conditional GAN (ACGAN), AdaFlow

facilitates flexible data imputation, adapting to various modalities and downstream tasks without retraining. Experiments show that AdaFlow significantly reduces inference latency by up to 79.9% and enhances accuracy by up to 61.9%, outperforming status quo approaches. Also, this method can enhance LLM performance to preprocess asynchronous data.

CCS Concepts

- Human-centered computing → Ubiquitous and mobile computing;
- Computing methodologies → Machine learning.

Keywords

Distributed multi-modal system, Non-blocking inference, Mobile applications, Affinity matrix

ACM Reference Format:

Fengmin Wu, Sicong Liu, Kehao Zhu, Xiaochen Li, Bin Guo, Zhiwen Yu, Hongkai Wen, Xiangrui Xu, Lehao Wang, and Xiangyu Liu. 2024. AdaFlow: Opportunistic Inference on Asynchronous Mobile Data with Generalized Affinity Control. In *ACM Conference on Embedded Networked Sensor Systems (SenSys '24)*, November 4–7, 2024, Hangzhou, China. ACM, New York, NY, USA, 13 pages. <https://doi.org/10.1145/3666025.3699361>

*Corresponding author: scliu@mail.nwp.edu.cn

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SenSys '24, November 4–7, 2024, Hangzhou, China

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM. ACM ISBN 979-8-4007-0697-4/24/11
<https://doi.org/10.1145/3666025.3699361>

1 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).

A core issue in distributed multi-modal systems is how to process the vast, continuously generated data streams from advanced sensors. 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.** Different input modalities (e.g., video vs. LiDAR) and dynamic network conditions lead to asynchronous data arrival at the fusion server. Also, low-power mobile sensors often lack clock synchronization. This may lead to delays (when waiting for slow data), or accuracy decline (when fusion proceeds without waiting). For example, in real-world autonomous driving scenarios with 6 cameras and 1 LiDAR, a 40ms asynchronous delay occurs in LiDAR data over a 100Mbps bandwidth, given 1 : 4 data size between LiDAR and camera data. This delay increases latency from 1s to 4.3s for 81 frames (in a 4s window) when waiting for slow data, whereas proceeding without waiting degrades accuracy by 49.7% (we defer more details in Sec. 2.1). However, mobile systems must minimize latency (e.g., $\leq 26ms$ per frame) to ensure safety and operational efficiency [10].

- **Heterogeneity.** Variations in the field of view, sensitivity, noise (e.g., lighting), and feature distribution of distributed multi-modal data affect the *consistency* and *complementarity* of fusion results. Improper management of these factors when deciding whether to discard or impute slow modalities can impact inference accuracy and latency. For example, using real-world autonomous driving dataset nuScenes [11], discarding 50% of the slow data which exhibits approximately 33% arrival delays, results in an 28.1% decrease in accuracy. Using KNN [12], a typical asynchronous data imputation method, reduces the issue but leads to a 9.7% accuracy drop and a 328% increase in latency (see Sec. 2.2).

As a separate note, cross-modal *consistency* and *complementarity*, collectively referred to as modality *affinity*, explore inference robustness and accuracy respectively by leveraging *modality-shared* and *modality-specific* information, respectively [13]. Modality affinity changes dynamically with input asynchronism and heterogeneity, variable across tasks, even with the same inputs (see Sec. 2.2). Given the above characteristics, existing synchronous (blocking) [1, 14, 15] and asynchronous (non-blocking) [16] techniques for distributed multi-modal inference face following challenges:

- **Challenge #1:** Most prior multi-modal methods optimize data *consistency* and *complementarity* to balance accuracy and latency. However, none offer a *quantitative* structured affinity analysis among

modalities for precise assessment and control. Also, most methods explore cross-modal affinity only *qualitatively* and under *static* or *ideal* noise-free conditions [17, 18].

- **Challenge #2:** Scheduling asynchronous and heterogeneous multi-modal data for low-latency inference is non-trivial, involving NP-hard multi-choice, multi-strategy optimization problems [17, 19, 20]. Moreover, a one-fit-all imputation module is essential yet challenging to handle *dynamic* and *diverse* input modalities, various subsequent tasks, and fluctuating asynchronous delays.

To address these challenges, we propose a shift to *opportunistic* inference, where the server performs clever inference as soon as partial asynchronous data are available. AdaFlow harnesses modality affinity to ensure robust data fusion at runtime. It is grounded in a fully computational method that models and optimizes inference based on modality affinity.

- **First**, drawing inspiration from [17], which explores *task* affinity using *unimodal* (visual) data during *learning*, we extend this concept to the *modality* level during *inference*, enabling dynamically adaptive assessments of diverse modalities. Furthermore, AdaFlow employs the analytic hierarchy process (AHP) to normalize matrix values, accommodating heterogeneity and asynchrony.

- **Second**, to enable low-latency and high-accuracy non-blocking inference that adapts to varying input modalities and types without retraining at runtime, we introduce a one-fit-all attention-based conditional generative network (ACGAN). This network adaptively embeds input features and imputes data, allowing for precise and timely control of modality affinity in opportunistic inference, paving the way for responsive mobile systems.

We evaluate the performance of AdaFlow on real-world 3D object recognition tasks and scenarios with diverse data heterogeneity and asynchrony. Results show a reduction of up to 79.9% in inference time with an improvement of up to 61.9% in accuracy. Especially, it displays the affinity matrix enables opportunistic inference based on different downstream tasks and even 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*. It eliminates waiting times for slow data and minimizes accuracy loss when excluding such data.
- We introduce AdaFlow, a system for opportunistic inference that adapts to asynchronous and heterogeneous data streams. It harnesses structured affinity control to enable adaptive fusion of dynamic data. Also, AdaFlow proposes an ACGAN to achieve precise and timely data imputation.
- Experiments show that AdaFlow outperforms existing a-/synchronous methods [12, 21–23] in trading off between inference accuracy and latency across various real-world mobile tasks, inputs, and scenarios.

2 Overview

2.1 Problem Analysis

To analyze and observe the motivation, we conducted the following experimental tests. We test a distributed multi-modal 3D object recognition system on nuScenes dataset [11]. nuScenes, 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. *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) [22], KNN (K-Nearest Neighbors) [12], and PCN (Point Completion Net) [23], along with the *synchronous* fusion method BM (Blocking Mechanism) [21], 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%).

2.2 Observations and Opportunities

In mobile applications, a distributed multi-modal data fusion system aims to optimize the tradeoff between inference accuracy and latency. A significant challenge with existing a-/synchronous methods (referenced in Sec. 7) is their failure to assess and manage quantitatively *modality affinity*, encompassing both modality consistency and complementarity, leading to delays or accuracy drop. Despite its importance, modality affinity remains a *black box*, exacerbated by the unpredictable asynchrony and heterogeneity in mobile contexts. To address this, we conduct specific tests and make the following observations.

A. 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~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~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~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.

B. 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

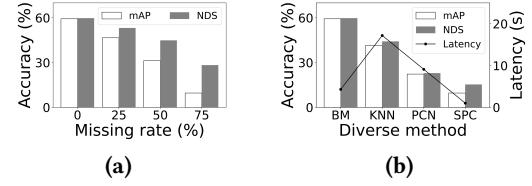


Figure 1: (a) Inference accuracy, (b) accuracy & latency of existing syn-/asynchronous methods, under varying missing rates of slow modality (*i.e.*, 3D point cloud).

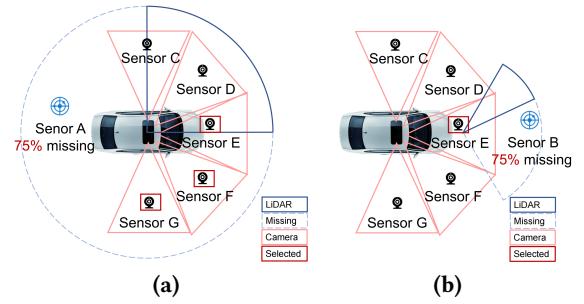


Figure 2: (a) Top LiDAR emphasizes complementarity and (b) Non-Top LiDAR emphasizes consistency.

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 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 [21]) results in significant variations in latency across three tasks, *e.g.*, 40ms for BOD, 25ms for BBS, 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 [22], KNN [12], PCN [23]) result in even more pronounced variations across three tasks. For example, when using PCN asynchronous 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.

3 Solution Overview

Drawing from the preliminary observations, we introduce AdaFlow to tackle the challenges of input modality *asynchrony* and *heterogeneity* in mobile distributed multi-modal inference systems (Sec. 1).

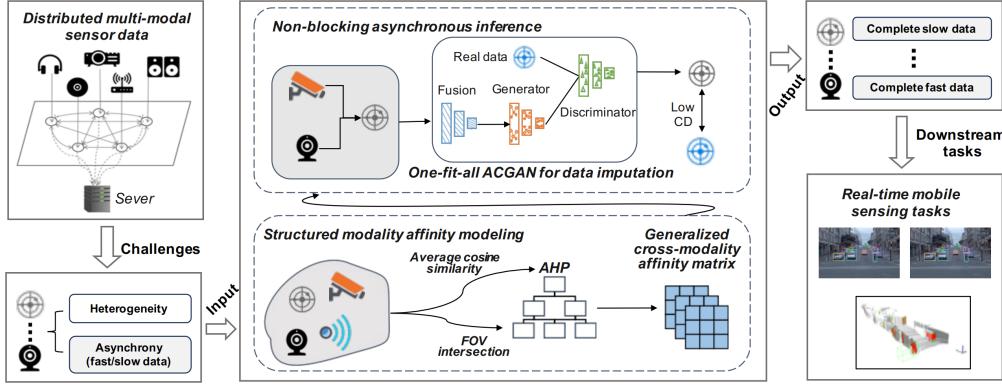


Figure 3: Workflow of AdaFlow.

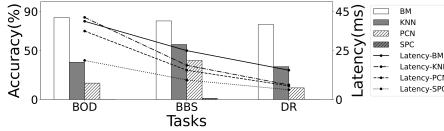


Figure 4: Inference latency and accuracy of BOD, BBS, DR tasks in existing a/synchronous methods.

Fig. 3 illustrates the architecture of AdaFlow (Fig. 16 shows the idea of AdaFlow), comprising two primary modules: Structured Modality Affinity Modeling and Non-blocking Asynchronous Inference. *First*, the *Structured Modality Affinity Modeling* module, detailed in Sec. 4, addresses *Challenge #1* by constructing a *generalized* cross-modality affinity matrix. This affinity matrix effectively manages data asynchrony and heterogeneity, enabling dynamic assessments of diverse input modalities. *Second*, AdaFlow confronts *Challenge #2* within the *Non-blocking Asynchronous Inference* module (Sec. 5). AdaFlow introduces a one-fit-all affinity attention-based conditional GAN (ACGAN), managing various input/output modalities without retraining. It ensures real-time, accurate inference via dynamic modality fusion, using a lightweight dynamic programming approach to select optimal affinity sub-graphs based on the affinity matrix. In real-world deployment, short delays from slow data may make a simple wait-and-process approach sufficient, without requiring modality affinity optimization.

4 Structured Modality Affinity Modeling

4.1 Generalized Modality Affinity Matrix

As valid in Sec. 2.2, distributed multi-modal data in mobile context often vary in views and arrival time, necessitating a *generalized* modality affinity matrix that accommodates diverse *tasks* alongside dynamic input data.

Preliminary of the data consistency and complementarity. *Consistency* refers to shared information across modalities, reinforcing fusion accuracy and reliability. *Complementarity* highlights unique, modality-specific insights. Both are crucial for robust multimodal fusion, balancing shared and unique data to optimize accuracy and latency.

Limitations of Prior Arts. Existing methods have explored data/task affinity matrices [17, 24, 25]; but they mainly focus on *single-view* scenarios [24] or *unimodality* [17, 25], failing to address the *dynamic asynchrony* and *heterogeneity* common in mobile contexts (see Sec. 1). For instance, ReID [24] uses triplet loss for data affinity in infrared cameras, while a vehicle-road system [25] examines bidimensional point clouds, and [17] studies task affinity in unimodal visual contexts. These methods, tailored to *specific* tasks, are inefficient for real-time mobile context due to the need for extensive customization. The key challenge is creating a *generalized* cross-modality affinity matrix to manage heterogeneity, asynchrony, and varied views in the mobile context.

In response, we employ t-SNE, a method known for capturing local structures and nonlinear relationships [26], to project heterogeneous modalities (of unlimited types) into a high-dimensional feature space. We then reduce these dimensions while preserving the spatial information by calculating the average cosine similarity at the feature level. We utilize cosine similarity at the *feature* level to measure modality affinity because it demonstrates stability despite data heterogeneity (*i.e.*, varied modalities) and asynchrony (*i.e.*, different rates of missing data). For instance, the average cosine distance between 3D point cloud and RGB image stands at 0.89192. Using features instead of raw data also reduces the size of affinity parameters, facilitating faster processing. This effectively reflects the consistency and complementarity between diverse input modalities.

Specifically, we utilize t-SNE [26] to map the Euclidean distances between high-dimensional data points into *joint probabilities* that represent their similarities. This method excels at capturing the local structure and nonlinear relationships within the data, which are essential for information-lossless dimensionality reduction. The degree of *similarity* between data points indicates their distance in joint probabilities, where a lower similarity implies a greater distance and higher *complementarity*. Thus, we can leverage such cosine similarity as a variable to represent *affinity*. In detail, the modality affinity relationship quantified by the joint probability p_{ij} , is calculated as below (line 2 in Algorithm 1):

$$p_{ij} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_k - x_i\|^2 / 2\sigma_i^2)} \quad (1)$$

where σ_i is the Gaussian variance centered on data point x_i . Given the redundancy of sensor data in high-dimensional space, we compress it to a lower dimension to enhance resource efficiency. The pair-wise affinity in this reduced space, q_{ij} (line 5), is computed as follow:

$$q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq l} \exp(-\|y_k - y_l\|^2)} \quad (2)$$

To minimize information loss during the above dimensionality reduction process, we minimize the Kullback-Leibler (KL) [27] divergence, a measure commonly used to assess the information loss between two probability distributions. Specifically, we minimize the KL divergence between p_{ij} and q_{ij} , which represent the high- and low-dimensional distributions of data point pairs, respectively. The optimization technique used in t-SNE is gradient descent, effectively minimizing the total KL divergence across data point pairs. The cost function C for this optimization is as follows:

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \quad (3)$$

where P is the affinity value in the high-dimensional space, Q is the corresponding affinity value in the low-dimensional space. The gradient formula is presented below:

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j). \quad (4)$$

Thus we have iteration as follows:

$$y^{(t)} = y^{(t-1)} + Adam(\frac{\partial C}{\partial y}) \quad (5)$$

where $y^{(t)}$ is the solution at iteration t , *Adam* represents Adam optimizer [19]. Upon obtaining the low-dimensional mapping of each modality at time t , we can readily calculate the average cosine similarity for each pair-wise modalities using t-SNE, as outlined in Algorithm 1.

We demonstrate the above process using Waymo [28] dataset, real-world autonomous driving data that includes three sensors: the front LiDAR, front camera, and left-side camera. They experience asynchrony due to varying data volumes. We perform t-SNE dimensionality reduction and visualization for the front LiDAR paired with the front camera, and separately for the front LiDAR with the left-side camera, with color-coded mappings to distinguish sensor data. The mapping of front LiDAR and front camera data shows a predominance of blue points—indicative of the LiDAR’s superior spatial data capture—alongside a common trend with green points. Typically, the threshold is set to the median in samples, which for the Waymo dataset is experimentally set around 0.06. Values above this indicate high consistency, while those below means high complementarity. The cosine similarity of 0.10808 between these sensors, as shown in Fig 5a, highlights their significant consistency and overlapping fields of view. Conversely, Fig 5b illustrates the mapping between the front LiDAR and the left-side camera, revealing a lower correlation with a cosine similarity of 0.03346. This indicates that these modalities provide complementary information due to differing views.

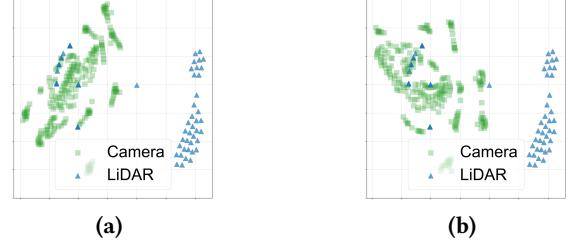


Figure 5: Visualization of (a) front LiDAR & front camera features and (b) front LiDAR & left camera features.

Algorithm 1 Generalized Cross-modal Affinity Matrix

Require: data set $X_k = \{x_{k1}, x_{k2}, \dots, x_{kn}\}$, number of iterations T , average cosine similarity S .

Ensure: low-dimensional data representation $y_k = \{y_{k1}, y_{k2}, \dots, y_{kn}\}$.

- 1: **for** $k = 0$ to T **do**
- 2: Compute pairwise affinities p_{ij} (using Equation 1)
- 3: Sample initial solution $y_k^{(0)} = \{y_{k1}^{(0)}, y_{k2}^{(0)}, \dots, y_{kn}^{(0)}\}$ from $\mathcal{N}(0, 10^{-4})$
- 4: **for** $t = 1$ to T **do**
- 5: Compute low-dimensional affinities q_{ij} (using Eq. 2)
- 6: Compute gradient $\frac{\partial C}{\partial y}$ (using Eq. 4)
- 7: Set $y_k^{(t)} = y_k^{(t-1)} + Adam(\frac{\partial C}{\partial y_k})$
- 8: $S = \text{Cos}(y_0^{(t)}, y_1^{(t)})$

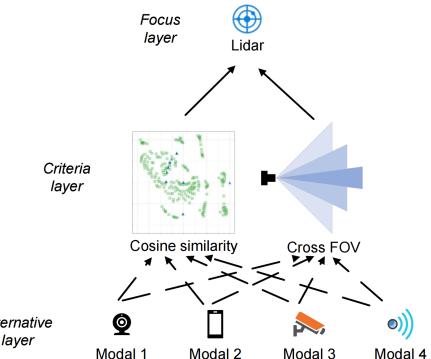


Figure 6: Affinity matrix normalization using AHP.

4.2 Normalization using Analytic Hierarchy Process (AHP)

In mobile systems equipped with multiple modality types and volumes, integrating the affinity measure between each pair of modalities into a unified framework is intractable.

Existing methods primarily model features within the same dimension [17, 24] or targeting specific characteristics like locations, semantics, sizes, and spatial distribution [25, 29, 30]. For instance, ReID [24] employs normalized Euclidean distance for affinity matrix elements within the same dimension. VIPS [25] examines node-to-node and edge-to-edge relationships in point cloud data. However,

Table 1: The scale of importance between sensors.

Intensity of importance $X_i(X = A, B)$ on an absolute scale	Definition
$x_{ij} = 1$	Equal importance between sensor i and j
$x_{ij} = 3$	Moderate importance of sensor i over sensor j
$x_{ij} = 5$	Essential or strong importance of sensor i over sensor j
$x_{ij} = 7$	Very strong importance of sensor i over sensor j
$x_{ij} = 9$	Extreme importance of sensor i over sensor j
$x_{ij} = 2, 4, 6, 8$	Intermediate values between the two adjacent judgments
Reciprocals	If there is an established importance between i and j , then the importance of j to i is the reciprocal of the x_{ij} .

they do not effectively extend to multi-modal problems. Normalization techniques including Quality-aware Multi-modal Fusion [18] and taskonomy [17] also exhibit limitations in addressing the system asynchrony (*i.e.*, varying missing rates of partial data) and heterogeneity (*i.e.*, modality type) challenges of distributed modality.

To effectively manage the diversity and dynamics of multi-modal fusion, we utilize the Analytic Hierarchy Process (AHP) to *normalize* affinity values. Normalization in this context means adjusting values measured on different scales to a notionally common scale. Although obtaining pair-wise modalities' affinity using Algorithm 1 is a straightforward procedure, applying this algorithm globally (across all data and modalities) and incorporating every single pair-wise modalities' affinity into a unified framework transforms the problem into a complex *graph programming* issue. Since the problem reduces to the vertex cover problem [31], it is NP-hard. In particular, we define the problem as Q , with a reduction algorithm f such that $Q = f(G(v_i, k))$, where $G(v_i, k)$ is a solution to the vertex cover problem. Algorithm f treats modalities as vertices v and their affinities as weighted edges in a graph. Setting $k = 1$ identifies the global optimal solution. Finding a subset of vertices based on weights has a time complexity of $O(n)$, meaning f runs in $O(n)$ time.

We observe that the complexity of the problem and the fact that sub-optimization is more resource-efficient, we are not aiming for a globally optimal solution. Instead, we can start with slow data for planning, which reduces the complex problem to a multi-objective programming problem [32] that is more suitable for AHP. In particular, our designed AHP quantitatively standardizes modality affinity and replaces subjective expert judgments with *empirical data*, thus reducing the controversies often associated with traditional AHP. The affinity matrix in our AHP 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. Initially, we intend to use mean cosine similarity as the sole affinity indicator; however, to mitigate potential biases in practice, we incorporate FOV intersection as an auxiliary measure.

To illustrate how the Analytic Hierarchy Process (AHP) is used to establish an affinity matrix, let's consider a simple example. Our AHP process consists of three layers, resulting in two layers of consistency matrices (conformity test matrices) named A and $\{B_1, B_2\}$. 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 detailed in Tab. 1. In this example, matrix A is straightforward, consisting of two rows and two columns. C_1 represents average cosine similarity and C_2 represents the intersection on the Field Of View (FOV). We assign $a_{12} = 7$ and $a_{21} = \frac{1}{7}$, emphasizing that FOV serves as an auxiliary adjustment rather than a primary measure, reflecting its lesser importance in this context. a_{ij} varies with the data signal-to-noise ratio. Increasing a_{21} and decreasing a_{12} can help correct bias.

$$A = \begin{matrix} C_1 & C_2 \\ C_1 & \begin{bmatrix} 1 & 7 \\ \frac{1}{7} & 1 \end{bmatrix} \\ C_2 & \end{matrix}$$

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, thus excluding subjective expert judgment. For example, with one Focus sensor and three Alternative sensors $\{sensor_1, sensor_2, sensor_3\}$, the matrix B_i is a 3×3 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.

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

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. By combining these vectors, we form the normalized affinity matrix.

5 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.

5.1 One-fit-all Data Imputation

Why conditional GAN (CGAN). A conditional generative adversarial network (CGAN) becomes viable for AdaFlow to predict missing or delayed input data across different modalities. CGANs excel in data imputation due to three key advantages : *i*) CGANs incorporate conditional inputs that govern the attributes of the generated data, aligning it more closely with the specific target distribution [33]. *ii*) CGANs excel in managing high-dimensional data and complex modalities. This is particularly beneficial for serial data streams [34]. *iii*) Shallow imputation methods treat each missing data independently, often leading to inconsistencies [35].

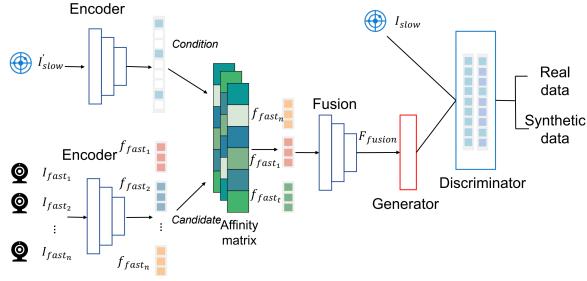


Figure 7: ACGAN-based non-blocking inference.

While CGANs learn the intrinsic structure and correlations within data, ensuring imputations are highly consistent with the actual distributed multi-modal distribution.

Preliminary of CGAN. A CGAN consists of two interlinked blocks: the generator G and the discriminator D . The generator learns to create data that mimics the target distribution from a latent space, while the discriminator evaluates the authenticity of samples generated by G , providing feedback that improves G 's outputs. These models engage in dynamic competition, continually adjusting to better simulate or identify real data. When conditional generation is required, G can incorporate specific data conditions to tailor the output, utilizing inputs from various data streams to enhance the completeness and relevance of its predictions.

However, due to *dynamic asynchrony* and *heterogeneity* in mobile sensing patterns, directly applying conditional GAN to a distributed multi-modal system is often ineffective. Traditional CGANs typically assume *synchronous* inputs with fixed input volume and type, which is not the case in mobile distributed systems where, for example, two types of LiDAR require different camera characteristics for G .

To tackle these challenges, we present an affinity-based conditional GAN (ACGAN), which selectively incorporates diverse amounts and types of modality as inputs. Unlike [16], which only accepts input from modality I_{fast_1} to generate slow data I'_{LiDAR} , our approach is one-fit-all, capable of accepting various inputs (e.g., any combination of inputs I_{fast_1}, I_{fast_2} , or I_{fast_3}) to generate data for slow data I'_{LiDAR} . As depicted in Fig. 7, the ACGAN integrates an attention-based fusion mechanism. It computes an attention matrix through *affinity calculation* and utilizes this matrix to perform a weighted fusion of features. This process selectively fuses appropriate fast and slow data, optimizing data fusion.

Specifically, as shown in Fig. 7, the ACGAN has an attention-based fusion, which generates an attention matrix through *affinity calculation* and uses the matrix to perform the weighted fusion of features to select appropriate fast data flows and slow data flows for fusion. Using LiDAR and cameras as examples, we extract camera features $F_{camera_1}, F_{camera_2}, \dots, F_{camera_i}$ from the affinity matrix h and LiDAR features F_{LiDAR} from incomplete data I'_{LiDAR} through a fusion layer g . This results in combined features F_{fusion} , which serve as inputs for the generator G to generate synthetic, complete LiDAR data I^*_{LiDAR} , replicating the characteristics of the full LiDAR dataset I_{LiDAR} .

Training of ACGAN. The training objective for ACGAN consists of two components: the conditional ACGAN loss \mathcal{L}_{CGAN} , and

a loss function for slow data stream characteristics \mathcal{L}_* , which measures the discrepancy between modalities. Inputs to the generator G include complete fast data stream $\{I_{fast_1}, I_{fast_2}, \dots, I_{fast_n}\}$ and the incomplete slow data stream I'_slow . This setup enables G to generate the synthetic slow data I^*_{slow} . The ACGAN loss function is formulated as follows:

$$\{F_{fast_1}, F_{fast_2}, \dots, F_{fast_n}\} = f_{fast}(\{I_{fast_1}, I_{fast_2}, \dots, I_{fast_n}\}) \quad (6)$$

$$F_{slow} = f_{slow}R(I'_slow) \quad (7)$$

$$F_{fusion} = g(h(\{F_{fast_1}, \dots, F_{fast_n}\}), F_{slow}) \quad (8)$$

$$I^*_{slow} = G(F_{fusion}) \quad (9)$$

$$\begin{aligned} \mathcal{L}_{CGAN} &= \mathbb{E}_{I_{slow}}[\log D(I_{slow})] \\ &+ \mathbb{E}_{I^*_{slow}}[\log(1 - D(I^*_{slow}))] \end{aligned} \quad (10)$$

where the generator G seeks to minimize Equ. 10, while the adversarial 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:

$$X = \{x_i\}_{i=1}^N Y = \{y_i\}_{i=1}^M, X = I^*_{LiDAR} Y = I_{LiDAR} \quad (11)$$

$$\mathcal{L}_{CD} = \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 + \sum_{y \in Y} \min_{x \in X} \|y - x\|_2^2 \quad (12)$$

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:

$$\mathcal{L} = \arg \min_G \max_D \mathcal{L}_{CGAN} + \lambda_{CD} \mathcal{L}_{CD} \quad (13)$$

where λ_{CD} is a hyper-parameter to tune the data fitting and model complexity. As a separate note, in autonomous driving, LiDAR data is generally slower than camera data. For instance, any combination of cameras I_{fast_1}, I_{fast_2} , and I_{fast_3} (with asynchrony) can generate LiDAR data I'_{LiDAR} , with the loss function incorporating point cloud. If the downstream task is fixed, asynchronous data can fill in for imputing the slower LiDAR without retraining. The loss function can also be adjusted for other modalities in different tasks.

5.2 Affinity-aware Sub-graph Selection

Existing data imputation methods [36, 37] predefine the type and number of modalities for data fusion and imputation, assuming fixed consistency and complementarity based on learned sensing patterns. However, these methods are unsuitable for mobile systems characterized by dynamics [16]. Studies [38, 39] reveal potential drawbacks. Specifically, [38] shows that adding more modalities without key ones can introduce noise and redundancy, degrading performance. Similarly, [39] reports that not all modalities contribute positively, with some introducing unnecessary noise. To address this, AdaFlow selectively chooses modalities that enhance fusion for better imputation (as we will show in Sec. 6.5.1.).

For example, AdaFlow uses real-world collected data in autonomous driving assistance, utilizing two LiDAR sensors and five cameras. 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. The problem can be described as a *decision graph*: given a set of heterogeneous modalities $M = \{m_1, m_2, \dots, m_N\}$ 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 $m_{fi} \rightarrow m_{sj} = F(M)$. For $F(M)$, we assess complementarity and consistency using the intersection over union between modalities. The intersection over union between modalities is defined as:

$$v_{ij} = \frac{\text{FOV}_{m_{sj}} \cap \text{FOV}_{m_{fi}}}{\text{FOV}_{m_{sj}} \cup \text{FOV}_{m_{fi}}} \quad (14)$$

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

$$m_{fi} \rightarrow m_{sj} = \begin{cases} \text{Consistency}, & \exists v_{ij} = 0 \\ \text{Complementarity}, & \nexists v_{ij} = 0 \end{cases} \quad (15)$$

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

$$A = \sum_{i=1}^j \sum_{l=1}^k a_{ij} \quad (16)$$

Where the value of k as shown below:

$$k = |\mathcal{V}_j| \times r \quad (17)$$

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

$$k = \begin{cases} k, k \neq 0 \\ 1, k = 0 \mid \mathcal{V}_j \mid \neq 0 \\ 0, k = 0 \mid \mathcal{V}_j \mid = 0 \end{cases} \quad (18)$$

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, AdaFlow 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. 4.2, 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.

6 Experiments

6.1 Setups

Implementation. AdaFlow 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. 4.2), we designate LiDAR as the primary modality and cameras as secondary modalities. For the ACGAN architecture (Sec. 5.1), 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 generator 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 [11, 40, 41].

- **BEVFusion** (T_1) is a real-world multi-modal 3D object recognition task using a bird's-eye view. The **dataset** (nuScenes [11]) 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 [40] 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)** [42] to categorize driver behaviors into 12 classes using the Drive&Act dataset [41], which comprises 6 infrared camera views (fast data) and 1 standard camera view (slow data).
- **LLM-based event recognition** (T_4) employs **LLM** for audio-visual event recognition across 28 categories using the AVE dataset [43] 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) [21]: the inference pipeline waits until all input data arrive. It provides an upper-bound accuracy.
- **Asynchronous fusion** (non-blocking mechanism):
 - Sparse Point Cloud (SPC) [22]: discards the slow data for inference which is the fastest method but exhibits the lowest accuracy.

- PCN [23]: 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 [12]: is a traditional imputation method, which classifies an input by majority class among its K nearest neighbors to selectively discard data.
- Traditional CGAN [44]: has the same imputation network structure as AdaFlow, but selects fixed or random fast data to impute the slow data. We use it to prove the effectiveness of the affinity matrix.

6.2 Performance Comparison

We evaluate the inference accuracy and latency of AdaFlow against four baseline methods (BM, SPC, PCN, and KNN) 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 nuScenes with 100Mbps.

Fig. 8 shows the results. *First*, AdaFlow exhibits the best trade-off between inference accuracy and latency compared to the baselines. *Second*, AdaFlow outperforms the three asynchronous fusion methods (SPC, PCN, and KNN) under various missing rates of slow data. For example, as shown in Fig. 8b, with 75% missing rate in BEV, AdaFlow demonstrates accuracy improvements of 45.4%, 32.7%, 13.3% compared to SPC, PCN, and KNN, respectively. *Third*, with acceptable accuracy, AdaFlow achieves the lowest latency compared to other baselines. As shown in Fig. 8d, with 50% missing rate of slow LiDAR data in PointPillars, AdaFlow achieves 44.5%, 78.7%, 70.6% latency reduction compared to BM, KNN, and PCN. Besides, AdaFlow improves 62% accuracy compared to SPC with only 1s slower than SPC for 81 frames.

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

6.3 AdaFlow’s Data Imputation Performance

We evaluate AdaFlow’s data imputation module on BEVFusion (T_1) and PointPillars (T_2) under different data missing rates of slow data. *First*, AdaFlow efficiently prevents significant accuracy decline at varying slow data missing rates. As shown in Fig. 9, AdaFlow suffer only 4.2%, 3.4%, and 4.4% 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*, AdaFlow is more necessary when suffering from *extreme* slow data missing. As shown in Fig. 10, AdaFlow improves 66% mAP and 55.7% NDS compared to without data imputation under data missing rates at 75%. Comparatively, without AdaFlow’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), AdaFlow maintains system effectiveness by preventing accuracy declines. Particularly in extreme asynchronous cases, AdaFlow avoids drastic accuracy drop.

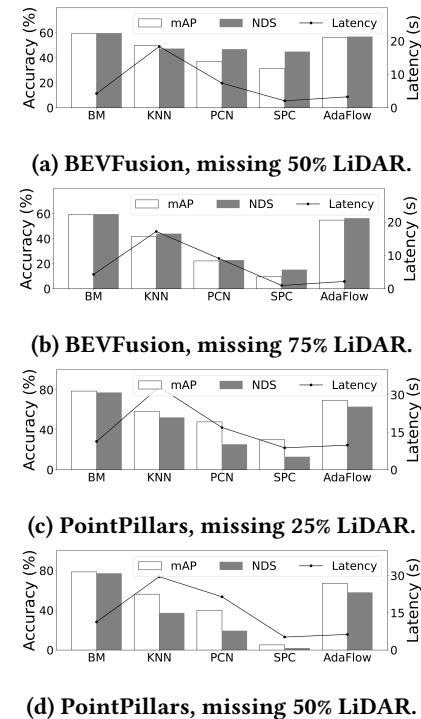


Figure 8: Comparison of accuracy and inference latency between AdaFlow and baselines in two tasks, under varying missing rates of slow data (*i.e.*, lagging delay).

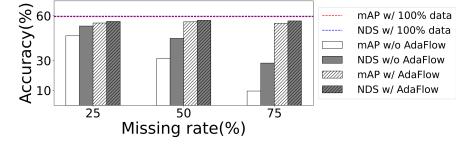


Figure 9: AdaFlow’s imputation performance in BEVFusion under different missing rates of slow data.

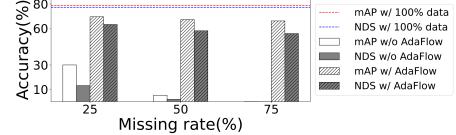


Figure 10: AdaFlow’s imputation performance in PointPillars under different missing rates of slow data.

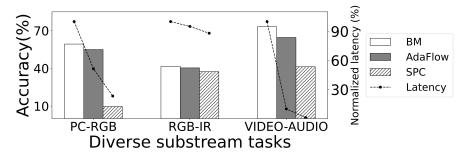


Figure 11: Performance over diverse downstream tasks.

6.4 Generalizing to Diverse Input Data

We evaluate AdaFlow’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. 11. *First*,

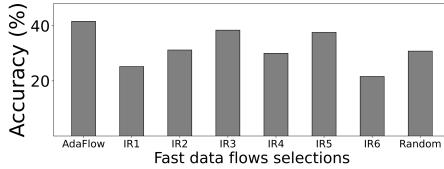


Figure 12: Comparison between AdaFlow and CGAN.

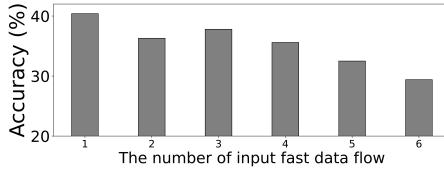


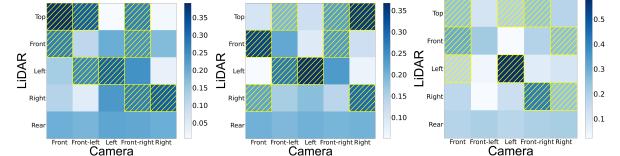
Figure 13: Performance on diverse modality numbers.

AdaFlow can adapt to various multi-modal tasks. In detail, on T_1 , T_3 , and T_4 , AdaFlow achieve 45.4%, 3%, and 23.32% accuracy improvements compared to SPC, a typical asynchronous baseline. *Second*, AdaFlow performs best when the data volume ratio between fast and slow modalities is large. As shown in Fig. 13. For data ratios of 1:4, 8:9, and 1:70, AdaFlow reduces latency by 44.5%, 2%, and 90%, respectively, compared to BM, a synchronous method with upper-bound accuracy. These gains are attributed to the one-fit-all ACGAN for dynamic data imputation and leverage a class attention mechanism for maximum data imputation.

6.5 Performance of Affinity Matrix

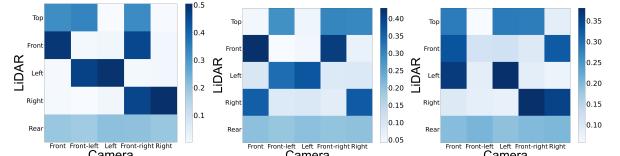
6.5.1 Validity Verification. We test AdaFlow’s affinity matrix on LLM task (T_3) under 10% data missing rates of slow data. *First*, AdaFlow achieves higher accuracy than CGAN’s 6 fixed selections, as shown in Fig. 12. AdaFlow’s accuracy is 40.6% and 6 CGAN’s 6 fixed selections is 36.3%, 37.8%, 39.6%, 37.5%, 39.4% and 35.4%, respectively. The inference derived by AdaFlow’s generalized affinity matrix is also higher than CGAN’s random selection (37.3%). *Second*, AdaFlow achieves higher accuracy with just one fast modality input, as shown in Fig. 13, while accuracy drops to 29.4% 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. 5.2.

6.5.2 Comparison to Monte Carlo. We compare the affinity matrix outperformed by AdaFlow and Monte Carlo in Fig. 14 and Fig. 15. 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 [45] 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. *First*, AdaFlow’s affinity matrix accurately captures the relationships between modalities. As shown in Fig. 14a and Fig. 15a, AdaFlow’s affinity matrix provides judgments on the relationships between modalities similar to the baseline, which is quantified as 0.89 using cosine similarity. *Second*, AdaFlow’s affinity matrix can dynamically adapt to changes in inter-modal relationships. As shown in Fig. 14,



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

Figure 14: Visualization of AdaFlow’s affinity matrix.



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

Figure 15: Visualization of expected affinity matrix.

the same modal settings exhibited different inter-modal relationships at different times, yet AdaFlow consistently provided similar predictions, *i.e.*, 0.89, 0.94, and 0.91 for the first frame, 10-th frame, and 20-th frame.

6.6 Case Study

Fig. 16 illustrates the latency breakdown in conventional end-to-end multi-modal object detection systems for autonomous driving, categorized into data preprocessing (e.g., handling or imputing slow data) and executing inference. AdaFlow enhances system performance by utilizing affinity-aware data imputation for preprocessing, thereby reducing the waiting time associated with slow data and improving overall detection responsiveness. To evaluate AdaFlow, we generated a 10,000-frame autonomous driving dataset using the Carla[46] autopilot simulator. As demonstrated in Fig. 17, the system processes inputs from six camera views (fast data) and one LiDAR view (slow data). We benchmark AdaFlow against two asynchronous methods, *i.e.*, KNN and PCN, comparing data preprocessing 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. 18c shows the results. AdaFlow reduces the overall latency of the preprocessing phase from 0.346s, 0.246s, and 0.141s to 0.059s, compared to KNN, PCN and BM respectively. While when imputing data similar to those collected by real sensors (see Fig. 18a and Fig. 18b), AdaFlow achieves latency reductions of 2.2 ~ 5.7× than BM, PCN, and KNN. In practical urban settings, where vehicles commonly travel at 36 km/h, with such latency reduction, AdaFlow reduces the detection distance interval from 1.42m-3.46m to just 0.625m. This significant improvement enhances the timely detection of pedestrians and vehicles, helping accident prevention.

We deploy AdaFlow with three different bandwidths, *i.e.*, 50bps, 75Mbps, and 100Mbps on NVIDIA AGX Xavier, similar Mercedes in-car chips. As shown in Fig. 19, AdaFlow remains efficient even under low bandwidth. Compared to BM, it reduces latency by 31% at 50Mbps while keeping CD error within acceptable limits (*i.e.*, ≤ 0.00015). We also test AdaFlow’s energy cost on the AGX Xavier.

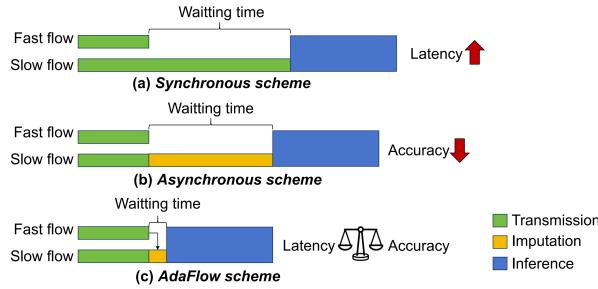


Figure 16: Paradigms of AdaFlow and baselines.

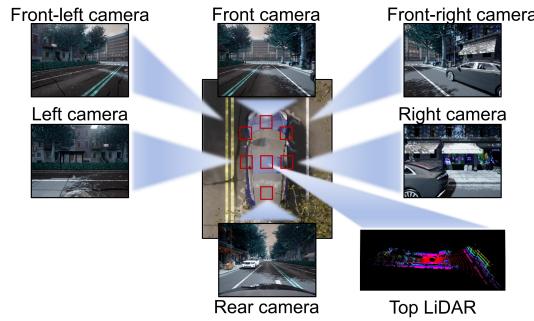


Figure 17: Illustration of seven sensors (six cameras and one LiDAR) on vehicle.

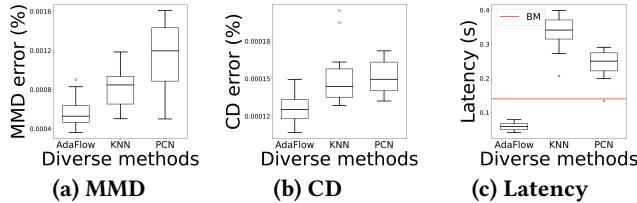


Figure 18: Comparison between AdaFlow and asynchronous baselines (KNN, PCN) in the self-collect dataset under 66% data missing rates of LiDAR.

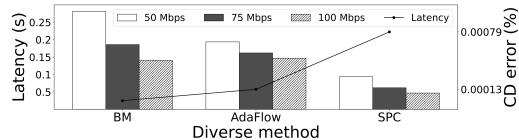


Figure 19: Performance with diverse bandwidths.

As shown in Fig. 20, AdaFlow's energy cost is around 22W, which is affordable.

7 Related Work

Multi-modal Inference in Mobile Applications. Multi-modal inference has demonstrated success across a range of mobile sensing tasks such as classification [14, 15, 47], object detection [48–53], segmentation [54–56], speech recognition [57], autonomous driving [11, 28, 40], and medical image segmentation [58]. For instance, the nuScenes dataset [11] illustrates how integrating multi-modal

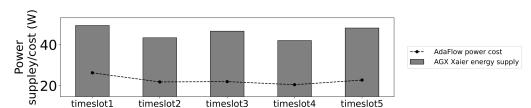


Figure 20: Energy cost of AdaFlow on AGX Xavier.

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 [59], 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.* [16], use predictive completion to reduce wait time but are limited by fixed input modality configurations. AdaFlow leverages cross-modal affinity for attention-based fusion, enabling efficient asynchronous inference across diverse input modalities, regardless of number or type.

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 [24], 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 [17], compute affinity from statistical data features without modifying the model. AdaFlow adopts the function-based approach for its flexibility. Unlike existing methods that are constrained by specific modalities, AdaFlow introduces a modality-agnostic affinity quantification method, enhancing cross-modal inference in dynamic, asynchronous settings.

8 Conclusion

To address data asynchrony and heterogeneity in distributed mobile environments, this paper introduces AdaFlow, which performs inference as soon as partial asynchronous data is available, improving efficiency. First, AdaFlow models and optimizes inference using a generalized modality affinity matrix, dynamically adapting to diverse inputs for effective multi-modal fusion. It also normalizes matrix values using the analytic hierarchy process (AHP). Second, AdaFlow employs a one-fit-all affinity attention-based conditional GAN (ACGAN) for high-accuracy, low-latency, non-blocking inference across various input modalities without retraining. Evaluation on real-world multi-modal tasks, AdaFlow significantly reduces inference latency and boosts accuracy. In the future, we can enhance adaptive affinity-based data imputation by leveraging the initial features of LLMs.

Acknowledgments

This work was partially supported by the National Science Fund for Distinguished Young Scholars (62025205) the National Natural Science Foundation of China (No. 62032020, 6247074224, 62102317).

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, Shiyuan 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 autopilot. <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] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liang, 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.
- [12] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.
- [13] 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.
- [14] 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.
- [15] Ajian Liu, Jun Wan, Sergio Escalante, 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.
- [16] 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.
- [17] 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.
- [18] Qingyang Zhang, Haitao Wu, Changqing Zhang, Qinghua Hu, Huazhu Fu, Joey Tianyi Zhou, and Xi Peng. Provable dynamic fusion for low-quality multi-modal data. In *International conference on machine learning*, pages 41753–41769. PMLR, 2023.
- [19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [20] 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.
- [21] 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.
- [22] 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.
- [23] 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.
- [24] 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.
- [25] Shuyaao Shi, Jiahui Cui, Zhehao Jiang, Zhenyu Yan, Guoliang Xing, Jianwei Niu, and Zhenchao Ouyang. Vips: Real-time perception fusion for infrastructure-assisted autonomous driving. In *Proceedings of MobiCom*, pages 133–146, 2022.
- [26] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.
- [27] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.
- [28] Pei Sun, Henrik Kretschmar, 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.
- [29] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.
- [30] Nitish Srivastava and Russ R Salakhutdinov. Multimodal learning with deep boltzmann machines. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.
- [31] Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover. *Annals of mathematics*, pages 439–485, 2005.
- [32] Michael TM Emmerich and André H Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, 17:585–609, 2018.
- [33] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [34] Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, et al. Multivariate time series imputation with generative adversarial networks. *Advances in neural information processing systems*, 31, 2018.
- [35] S Cheng-Xian Li, B Jiang, and B Marlin. Learning from incomplete data with generative adversarial networks, 2019, doi: 10.48550/ARXIV, 1902.
- [36] 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.
- [37] 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.
- [38] 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.
- [39] 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.
- [40] 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.
- [41] 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.
- [42] 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.
- [43] Yapeng Tian, Jing Shi, Bochen Li, Zhiyuan Duan, and Chenliang Xu. Audio-visual event localization in unconstrained videos. In *Proceedings of ECCV*, pages 247–263, 2018.
- [44] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- [45] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of AAAI*, volume 30, 2016.
- [46] 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.
- [47] Haifan 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.
- [48] 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.
- [49] Peng Sun, Wenhui 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.
- [50] 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.
- [51] Sicong Liu, Bin Guo, Ke Ma, Zhiwen Yu, and Junzhao Du. Adaspripg: Context-adaptive and runtime-evolutionary deep model compression for mobile applications. *Proceedings of UbiComp*, 5(1):1–22, 2021.
- [52] Lehao Wang, Zhiwen Yu, Haoyi Yu, Sicong Liu, Yaxiong Xie, Bin Guo, and Yunxin Liu. Adaevp: Edge-assisted continuous and timely dnn model evolution for mobile devices. *IEEE Transactions on Mobile Computing*, 2023.
- [53] 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.

- [54] Jinming Cao, Hancho Leng, Dani Lischinski, Daniel Cohen-Or, Changhe Tu, and Yangyan Li. Shapeconv: Shape-aware convolutional layer for indoor rgbd 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 Wengfeld, and Horst-Michael Gross. Efficient rgbd semantic segmentation for indoor scene analysis. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 13525–13531. IEEE, 2021.
- [57] Stavros Petridis, Themis 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] 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.
- [59] 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.