# StableSynthNet: Disentangled HyperNetworks for Enhanced On-device Multi-modal Model Generalization

**Anonymous ACL submission**

## Abstract

In the modern interconnected landscape, the proliferation of smart devices leads to the continuous collection of extensive and varied personal multi-modal data. This situation necessitates the development of sophisticated, personalized, and device-aware services. Traditional AI systems, mainly cloud-based, face considerable hurdles in adapting to the dynamic data flow between cloud services and devices. While HyperNetworks have enhanced performance and real-time processing over conventional fine-tuning approaches, they tend to be over-parameterized due to the underutilization of consistent data types. Our solution, StableSynthNet, is a novel system consisting of three components: Driver Contrastive Training, Template-Driver Extraction, and Offset-Driver Separation. This design uniquely separates the template parameter driver, which houses common data characteristics, from the offset parameter driver, where individual data specifics are stored. The resulting combined driver achieves an optimal mix of consistency and adaptability. Our extensive testing in the fields of video question answering and video retrieval has demonstrated the superior efficiency and effectiveness of StableSynthNet.

## 1 Introduction

In the current digital age, the widespread adoption of smart devices, encompassing everything from smartphones to the expansive Internet of Things network, is a fundamental aspect of daily life. These devices gather vast amounts of personal data in various forms such as text, images, and videos. This rich and varied data holds significant promise for providing highly personalized services. However, efficiently processing this ever-changing data to accurately reflect user preferences is a primary challenge (Howard et al., 2019; Sandler et al., 2018; Howard et al., 2017).

Currently, most AI systems operate in the cloud and struggle to adjust to the distinct data characteristics of each device due to the variable distribution of data between the cloud and individual devices. Therefore, to deliver high-quality, personalized services, AI systems must be designed to continuously evolve and adapt to the unique and fluctuating nature of user-specific data.

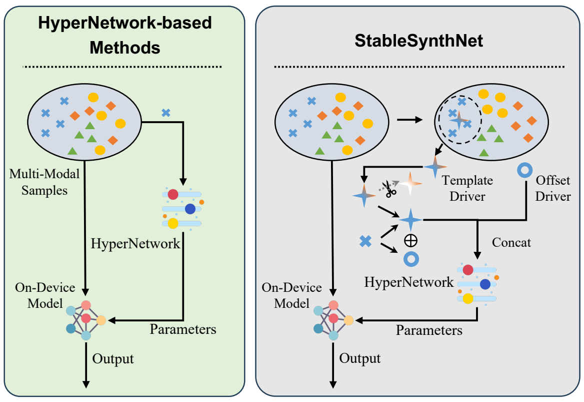

Figure 1: The pipeline of HyperNetwork-based Method and our proposed StableSynthNet.

A seemingly simple approach to bridge the gap between cloud-based AI systems and individual devices involves modifying the cloud model with data from these devices, addressing the disparity in data distribution. However, this method, known as fine-tuning adaptation (FTA), faces several challenges. FTA often requires extensive manual data labeling, leading to delays in the AI system's responsiveness. This problem is exacerbated by the complexities of interpreting multi-modal data, making FTA inefficient for real-time applications. Additionally, FTA is prone to overfitting, especially in devices with limited or highly specific data, which can degrade the model's performance across various devices (Chen and Wang, 2021; Li et al., 2020; Yuan et al., 2022a,b; Zhang et al., 2020, 2021, 2023b).

Recently, HyperNetwork-based methodologies (Dinh et al., 2022; Ha et al., 2017; Yi et al., 2023) have gained traction as a means to enhance model generalization across diverse data distributions. These methodologies dynamically generate model parameters tailored to each individual data sample's unique distribution. However, when facing real-time samples with intense distribution fluctuations, HyperNetworks exhibit significant performance variability due to over-parameterization, leading to inefficiencies and complexities in model deployment and scalability. Additionally, HyperNetworks require fine-grained features for optimal performance. Meanwhile, the high communication cost and slow speed between edge and cloud prevent the guaranteed upload speed of fine-grained features.

To effectively address this challenge, our work introduces StableSynthNet, which decouples the Parametric Driver used to generate model parameters in the HyperNetwork-based method into a Template Parametric Driver and an Offset Parametric Driver. This approach avoids the shortcomings of over-parameterization caused by HyperNetwork-based rapid generalization of device models.

The model is built on three basic pillars: **Driver Contrastive Training**, **Template-Driver Extraction**, and **Offset-Driver Separation**. **Driver Contrast Training** uses contrastive learning to move the Parametric Driver closer to the same type and farther from different types, improving separability. **Template Driver Extraction** focuses on creating diverse Template Parametric Drivers, each encapsulating homogeneous information from a sample category. This driver utilizes information from similar samples, which existing HyperNetwork-based methods ignore. **Offset-driver Separation** explores and establishes invariant representations in data. It dynamically updates the Template Parametric Driver to improve its representation capabilities and separates the Shift Parametric Driver to achieve stability and generalization compatibility.

Our work makes several key contributions:

- We propose a unique framework, StableSynthNet, that decouples model generalization from the pitfalls of over-parameterization, advancing HyperNetwork-based methodologies.

- The Template-Driver Extraction learns stable multi-modal representations for HyperNetwork generalization. Driver Contrastive Training ensures intra-class compactness, and Offset-Driver Separation explores invariant representations.

- Extensive experiments on video question answering and video retrieval tasks verify the efficiency and effectiveness of our method.

## 2 Related Works

**Cloud-device Collaboration.** In cloud-device collaboration, deep learning combines cloud resources with on-device processing for enhanced performance. Federated Learning, exemplified by FedAVG (Collins et al., 2022), faces scalability and efficiency challenges. Methods like Multi-Path Device Adaptation (Yan et al., 2022) improve on-device model performance using cloud-based samples, bridging centralized training and decentralized execution. DUET (Lv et al., 2023b) simplifies model adaptation by pre-training components in the cloud, reducing on-device computational demands. IDEAL (Lv et al., 2023a) extends parameter generation-based models to recommender systems, focusing on cross-domain recommendation (Zhang et al., 2023a) for better model generalization. However, HyperNetwork methods often face instability and delays due to over-parameterization, affecting training and deployment efficiency. To address this, we introduce StableSynthNet, designed to mitigate over-parameterization, ensuring stable model adaptation and improved efficiency. This enhances device model robustness and adaptability in dynamic environments, advancing cloud-device collaborative deep learning.

**Domain Adaptation.** Domain Adaptation (DA) transfers a network trained on a labeled source domain to a target domain, especially when their data distributions differ. Techniques like maximum mean discrepancy and correlation alignment reduce distribution discrepancies to improve generalization. Various methods, such as multi-stage (Zhang et al., 2019; Chen et al., 2021) and gradual transfer strategies (Chen et al., 2019), incrementally align domains. Curriculum-based strategies (Shu et al., 2019; Roy et al., 2021) structure adaptation by increasing task complexity. Source-Free Domain Adaptation addresses privacy concerns by operating without source domain data. DA has been applied to tasks like action recognition (Wang et al., 2023b), natural language processing (Mosallanezhad et al., 2022), and object detection (Ahmed et al., 2023; da Costa et al., 2022), demonstrating its broad applicability. In summary, DA methods align feature distributions to improve performance. We propose StableSynthNet, a novel approach for device model generalization, enhancing robustness and adaptability across diverse domains.

## 3 Method

### 3.1 Preliminary

**Problem Formulation.** For the on-Device Multi-modal Model Adaptation (DMMA) in the device-cloud collaboration system, we have access to a set of devices $\mathcal{D} = \{d^{(i)}\}_{i=1}^{\mathcal{N}_d}$, each device with its personal i.i.d multi-modal history samples $\mathcal{S}_{H^{(i)}}$ and multi-modal real-time samples $\mathcal{S}_{R^{(i)}}$ in current session, where $\mathcal{N}_d$ represents the number of devices. The goal of on-Device Multi-modal Model Adaptation is to generalize a trained cloud model $\mathcal{M}_g(\cdot; \Theta_g)$ learned from $\{\mathcal{S}_{H^{(i)}}\}_{j=1}^{\mathcal{N}_d}$ to each specific local device model $\mathcal{M}_{d^{(i)}}(\cdot; \Theta_{d^{(i)}})$ conditioned on real-time samples $\mathcal{S}_{R^{(i)}}$, where $\Theta_g$ and $\Theta_{d^{(i)}}$ respectively denote the learned parameters for the cloud model and the $i$-th device model:

$$\begin{aligned}\underbrace{\textbf{StableSynthNet}}_{\text{DMMA Model}} : \underbrace{\mathcal{M}_g(\{\mathcal{S}_{H^{(i)}}\}_{i=1}^{\mathcal{N}_d}; \Theta_g)}_{\text{Cloud Model}} \\ \rightarrow \underbrace{\mathcal{M}_{d^{(i)}}(\mathcal{S}_{R^{(i)}}; \Theta_{d^{(i)}})}_{\text{Device Model}}.\end{aligned} \quad (1)$$

Figure 2 illustrates the overview of our framework which consists of two modules to improve the generalization ability of the trained models on the device: (a) *HyperNetwork-based generalization* aims to learn a global benchmark model based on the history samples of all distributions and generate the network parameters for the distribution-specific device model based on the real-time device samples; (b) *StableSynthNet* standardizes the input for the HyperNetwork across various multi-model samples. (in Sec. 3.2).

**Model Pipeline.** There are three steps in the pipeline of our framework: (1) Training the cloud model $\mathcal{M}_g(.)$, including the HyperNetwork-based generalization module and the StableSynthNet module, with the history samples $\mathcal{S}_H$. (2) Uploading the real-time samples $\mathcal{S}_{R^{(i)}}$ from the device side to the cloud side. Then, the cloud side model $\mathcal{M}_g(.)$ generates the personalized parameters for the device model $\mathcal{M}_{d^{(i)}}(.)$. (3) With the model parameters passed from the cloud side, the device model $\mathcal{M}_{d^{(i)}}(.)$ is updated and makes the final prediction based on the input read-time samples.

**Multi-modal Feature Extraction.** We extract the multi-modal representation needed for the subsequent processes from the input examples. Specifically, given a real-time input video $V_r$ and a corresponding language query $Q_r$, we employ the DeiT (Touvron et al., 2021) as the visual feature extractor and the BERT-base language model (Devlin et al., 2019) as the text encoder, obtaining $F_v = \{F_v^i\}_{i=1}^{N_f}$, which represents the features of all video frames and $N_f$ is the video frame number, and the semantic feature $F_t$ of the query.

We adopt the spatial-temporal positional embedding and modality type embedding following (Wang et al., 2023a) to the extracted features $F_v$ and $F_t$ and get the corresponding embedded features $E_v = \{E_v^i\}_{i=1}^{N_f}$ and $E_t$. Then, the cross-modal fusion module $g(.)$ consisted of $t$ transformer layers is employed to fuse the visual features $E_v$ and the language feature $E_t$:

$$F_m = g(E_v, E_t), \quad (2)$$

where $F_m = \{F_m^i\}_{i=1}^{N_f}$ is the generated multi-modal features after the feature fusion.

**HyperNetwork-based Generalization.** Adapting the cloud model to real-time samples on a personalized device usually includes on-device fine-tuning. However, this can be challenging due to limited training samples and annotations on the device. To address this, we propose the HyperNetwork-based generalization, a novel solution implemented as a cloud service. This module processes images captured by the device, generating device-specific parameters. These parameters are meticulously crafted to adapt to the distinct data distributions encountered on each device.

Specifically, we further sample $D(D > 1)$ frames from $F_m$ randomly, and average the included $D$ frames of multi-modal features into the global representation $F_g$. Meanwhile, only one frame is then randomly sampled, and its multi-modal feature $F_m^i$ is uploaded from the device side to the cloud side. Then, we project the feature $F_m^i$ before it is further analyzed:

$$\Theta_d = f_h(f_p(F_m^i)), \quad (3)$$

where $f_p(.)$ represents the multi-layer perceptrons and generates real-time sample embedding for hyper-network (Dinh et al., 2022; Ha et al., 2017) $f_h(.)$. The generated $\Theta_d$ is the parameters for a single linear layer including both the linear weights and bias, which will be passed to the device model.

The device model $\mathcal{M}_d$ is updated with the model parameters $\Theta_d$ and makes the prediction $P_n$ for the

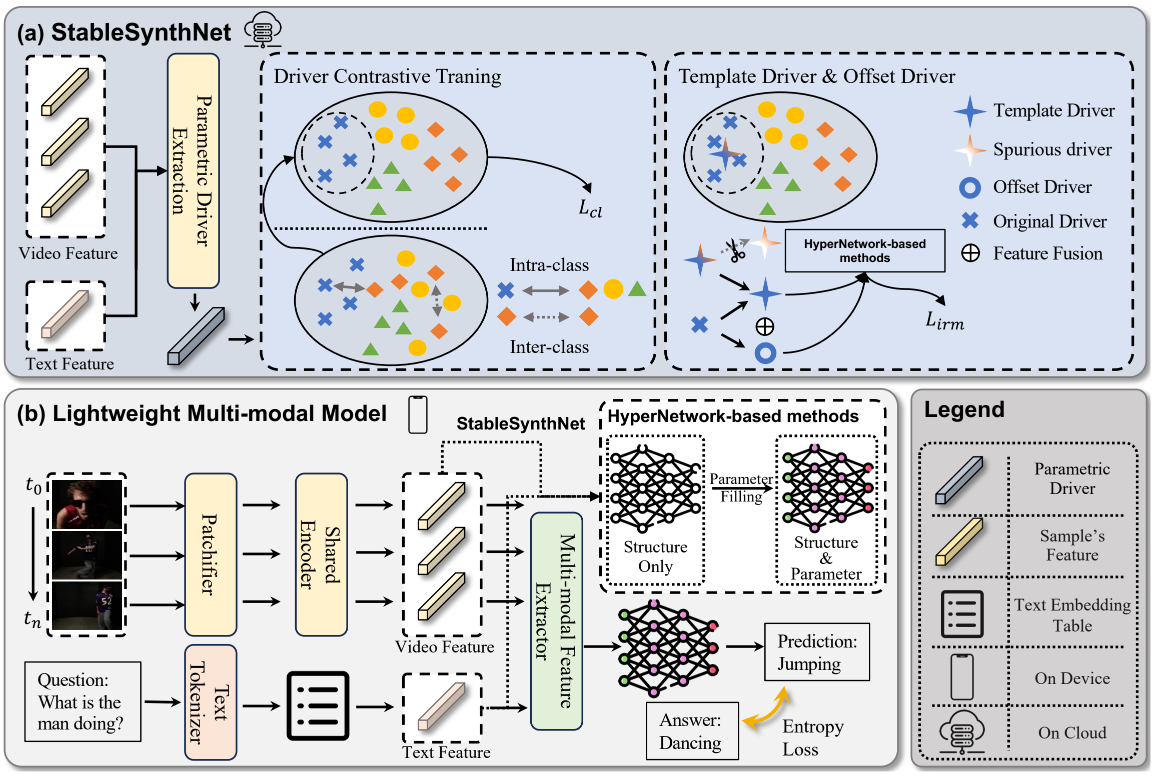

Figure 2: Overall pipeline of our proposed StableSynthNet, which includes Driver Contrastive Training, Template-Driver Extraction, and Offset-Driver Separation. StableSynthNet is deployed on the cloud and uniquely separates the template parameter driver, which houses common data characteristics, from the offset parameter driver, where individual data specifics are stored. Our approach achieves an optimal mix of consistency and adaptability.

input multi-model real-time sample:

$$P_n = \mathcal{M}_d(F_g; \Theta_d). \tag{4}$$

## 3.2 StableSynthNet

StableSynthNet is designed to adapt to diverse multi-modal tasks using specific data inputs. However, tasks like Video QA require transmitting large amounts of data (e.g., multiple frames or entire videos) to the HyperNetwork, leading to high communication costs and bandwidth requirements. To address this and improve HyperNetwork adaptability across multi-modal tasks, StableSynthNet standardizes the input format for compatibility.

With bandwidth limitations, an uploaded one-frame multi-modal representation can suffer significant distribution shifts. To mitigate this, we use template-driver extraction to maximize discriminative power between categories. By measuring invariance within each category and dynamically updating the multi-modal template, StableSynthNet ensures a consistent embedding space for accurate HyperNetwork-based generalization.

**Driver Contrastive Training.** During the training process, all history samples are directly saved on the cloud side and participated in the training process, following the setting of previous methods (Yao et al., 2021).

Formally, given a sample $s_i$ with category label $a_i$ in the batch $S = \{s_k\}_{i=k}^{N}$, we can construct its positive set $PS_i = \{s_k | a_i = a_k\}_{k \neq i}$ and negative set $PN_i = \{s_k | a_i \neq a_k\}_{k \neq i}$. With the training data, we use an InfoNCE loss to optimize the StableSynthNet:

$$L_{cl} = \sum_{\text{i=1}}^{\text{N}} -\log \frac{f_{pos}}{f_{pos} + f_{neg}}. \tag{5}$$

To further boost the sensitivity to different representations of categories, we additionally introduce an angular margin $m$ for the positive pairs. Formally, we formulate $f_{pos}$ and $f_{neg}$ as:

$$\begin{aligned} f_{pos} &= \sum_{s_j \in PS_i} e^{cos(\theta_{i,j}+m)/\mathrm{T}}, \\ f_{neg} &= \sum_{s_g \in NG_i} e^{cos(\theta_{i,g})/\mathrm{T}}, \end{aligned} \tag{6}$$

where T is a temperature hyper-parameter, $N$ is the batch size, $\theta_{i,j}$ is the arc-cosine similarity between the uploaded one-frame multi-modal features of $s_i$ and $s_j$.

**Offset-Driver Separation.** We propose the offset-driver separation to further promise the consistent representation of samples.

Specifically, we say that a consistent representation $\Phi : X \to H$ elicits an invariant predictor $\omega$ across the positive set $\varepsilon$ if there is an optimizer $\omega : H \to Y$ simultaneously optimal for all samples from the positive set. The learning objective can be formulated as:

$$\omega \in argmin_{\bar{\omega}:H\to Y} O^e(\omega, \Phi). \tag{7}$$

Eq. 7 tries to learn a feature representation from $\Phi(\cdot)$ that can induce an optimizer $\omega(\cdot)$ which is simultaneously optimal for all $e \in \varepsilon$. Thus, we propose the invariant representation regularization which can be formulated as:

$$L_{irm} = \sum_{i=1}^{N} \lambda Var(L^i), \tag{8}$$

where $L^i = \{L_{cl}(j) \,|a_i = a_j\}$ denotes the loss values of samples from the positive set, and $\lambda$ is a hyper-parameter. The minimization of variances of loss values encourages consistent representation learning for each category label (Arjovsky et al., 2020).

**Template-Driver Extraction.** Specifically, we specify the total template space $P_{type} \in \mathbf{R}^{V\times Q}$, where $V$ is the same as the size of multi-modal feature embedding. Then we dynamically update the template space depending on the degree of invariance during the driver contrastive training process.

Given a batch $S$, we can construct multiple positive sets $PS = \{ps_1, ps_2, ..., ps_P\}$ with different category labels, where $P$ is the number of different category labels in the batch $S$.

For every positive set in the $PS$ with category $a_i$, we obtain its average multi-modal representation embedding as follows:

$$H^{a_i}_{aver} = \frac{1}{N^{a_i}} \sum_{s=1}^{N^{a_i}} h_i, \tag{9}$$

where $N^{a_i}$ denotes the number of samples with category label $a_i$ in the batch $S$, and $h_i$ denotes the multi-modal feature of sample. We average the summation of all multi-modal representations with the same category label in the batch to get the average feature embedding.

With the help of observed invariant representation, we update the template space for category label $a_i$ with a moving average approach:

$$\begin{cases} P^{a_i}_{type} = \beta P^{a_i}_{type} + (1-\beta) H^{a_i}_{appr}, \\ H^{a_i}_{appr} = \underbrace{\frac{1}{\gamma Var(L^i) N^{a_i}}}_{Approach\ Speed} (H^{a_i}_{aver} - P^{a_i}_{type}). \end{cases} \tag{10}$$

where $H^{a_i}_{appr}$ is formulated as below, and $\beta$ and $\gamma$ are hyper-parameters:

During inference stage, in order to incorporate the various distribution shifts into the feature of the single video frame, we attempt to fuse the uploaded feature $F^i_m$ with category label $a_i$ and its template $P^{a_i}_{type}$.

We re-normalize the $P^{a_i}_{type}$ to have the same channel-wise mean and standard deviation as the $F^i_m$ (Luo et al., 2020). This process can be formulated as follows:

$$\Upsilon a(P^{a_i}_{type}, F^i_m) = \delta(P^{a_i}_{type}) \left( \frac{F^i_m - \delta(F^i_m)}{\gamma(F^i_m)} \right) + \gamma(P^{a_i}_{type}), \tag{11}$$

where $\delta(\cdot)$ and $\gamma(\cdot)$ denotes channel-wise mean and standard deviation operations, respectively.

Finally, the fused multi-modal feature is obtained by the following process:

$$F' = \lambda \Upsilon a(P^{a_i}_{type}, F^i_m) + (1-\lambda) F^i_m, \tag{12}$$

where the $\lambda$ is a hyper-parameter.

The feature $F'$ is applied to replace the global representation of the input sample $F^i_m$ defined in Eq. 3 for further prediction of the personalized model parameters.

## 4 Experiment

### 4.1 Tasks and Implementation Details

**Tasks.** We evaluate our method on three tasks: (1) **Video-text Retrieval:** We evaluate two sub-tasks including video-to-text retrieval and text-to-video retrieval. (2) **Open-ended Video-question Answering:** It requires answering questions according to the context of the video. The answers are originally in free-form natural language, but it is a common practice to convert the task to a classification task by representing the answer with a class label. (3) **Multiple-choice Video-question Answering:** Given a video with a query and 5 candidate captions, the task is to find the one that fits the query out of 5 possible candidates.

Table 1: Results of our proposed method in Open-ended Video QA task. We evaluate our method on three datasets: MSRVTT-QA, MSVD-QA, and TGIF. We adopt accuracy as the evaluation metric and the time delay is additionally measured to show the efficiency of our proposed method. "F-linear" denotes only fine-tuning the classifiers after the multi-modal feature extractor. "F-hyper" denotes only fine-tuning the classifiers and a simple hyper-network without an adaptive generator.

| Methods | MSRVTT-QA | | MVSD-QA | | TGIF | |
|---|---|---|---|---|---|---|
| | Accuracy | Time Delay | Accuracy | Time Delay | Accuracy | Time Delay |
| F-linear | 13.6 | $\geq$60000ms | 17.3 | $\geq$60000ms | 2.73 | $\geq$ 60000ms |
| Fine-tuning | 36.7 | $\geq$60000ms | 34.3 | $\geq$60000ms | 54.7 | $\geq$60000ms |
| F-hyper | 11.1 | $\geq$5.55ms | 6.34 | $\geq$3.71ms | 19.6 | $\geq$5.70ms |
| Ours | **36.9** | $\geq$5.55ms | **35.3** | $\geq$3.71ms | **55.5** | $\geq$5.70ms |

Table 2: Performance comparison (Accuracy and latency) between DUET and our method on three datasets. MSR. denotes MSRVTT, MSV. denotes MSVD, and TGI. denotes TGIF.

| Meth. | MSR. | | MSV. | | TGI. | |
|---|---|---|---|---|---|---|
| | Acc. | Lat. | Acc. | Lat. | Acc. | Lat. |
| DUET | 33.4 | 1.80s | 34.2 | 1.88s | 55.2 | 2.00s |
| Ours | 36.9 | 1.24s | 35.3 | 1.16s | 55.5 | 0.99s |

## 4.2 Datasets

For the open-ended Video QA task, we use the MSRVTT-QA dataset (Xu et al., 2017), MSVD-QA dataset (Xu et al., 2017) and the TGIF dataset (Li et al., 2016). For the multiple-choice Video QA task, we use the MSRVTT-QA dataset (Xu et al., 2017). For the video-text retrieval task, we experiment on the MSRVTT dataset (Xu et al., 2016).

## 4.3 Evaluation Metrics

Following previous works, we adopt accuracy for the open-ended Video QA task, and VR@K (video to text retrieval), TR@K (text to video retrieval) for the video-text retrieval task. For both of these two tasks, K is set to 1,2,5 respectively. Additionally, we calculate the number of learnable parameters in each model. Meanwhile, for the practical scenario of cloud-device collaboration, we also measure the time delay for the cloud-device communication process.

**Implementation Details.** We use the All-in-One-Ti model (Wang et al., 2023a) as our baseline, integrating DeiT (Touvron et al., 2021) as the visual backbone and BERT-base (Devlin et al., 2019) (using only its embedding layers) as the semantic encoder. Both the device and cloud models employ this multi-modal encoder. The parameter $D$ is set to 3. During training, we use the AdamW optimizer with a polynomial decay scheduler, a learning rate of 2e-5, and 10 epochs. For other modules, we freeze the adaptive generator parameters, use the AdamW optimizer with a learning rate of 1e-4, and train for 40 epochs. During training, $\lambda$ is set to 0.1, and the hyper-network's hidden layer size is 96 for Open-ended and Multiple-choice VQA, and 256 for Video-text Retrieval. During inference, all parameters are frozen except for the last few dynamic linear layers of the device model. The cloud model and the device model share the same multi-modal encoder parameters. The cloud generates dynamic parameters using the hyper-network for the device model to enhance generalization.

## 4.4 Performance Comparison

**Baseline Methods.** As the first to explore this field, we design three baseline methods to demonstrate the superiority of our proposed method. In Tab. 1, we present the fine-tuning approach, fine-tuning only the linear classifiers approach (F-linear), and fine-tuning both the linear classifiers and the hyper-network approach (F-hyper). (1) Fine-tuning approach: We add layers of MLPs after the multi-modal feature extractor and fine-tune the entire model. (2) F-linear: We freeze the parameters in the multi-modal feature extractor and fine-tune only the MLP layers. (3) F-hyper: We use a simple hyper-network without the adaptive generator and fine-tune both the MLP layers and the hyper-network.

**(1) Open-ended Video-question Answering.** For Open-ended Video-question Answering, responses are expressed in unrestricted natural language but often transformed into a classification problem by encoding answers as class labels. We incorporate layers of MLPs with a hidden layer dimension of 96 after extracting multi-modal features. The MLP's output layer dimension varies with the label size of

Table 3: Results of our method in Text-video Retrieval task. “VR@K" denotes the recall rate of video-to-text retrieval, and “TR@K" presents the recall rate of text-to-video retrieval.

| Methods | MSRVTT | | | | | |
|---|---|---|---|---|---|---|
| | VR@1 | VR@5 | VR@10 | TR@1 | TR@5 | TR@10 |
| F-linear | 2.0 | 7.7 | 14.1 | 3.1 | 10.1 | 16.9 |
| Fine-tuning | 4.8 | 17.6 | 27.9 | 6.2 | **20.7** | 31.8 |
| F-hyper | 2.6 | 8.6 | 14.7 | 2.7 | 11.4 | 17.6 |
| Ours | **5.8** | **20.2** | **31.2** | **6.4** | 20.4 | **33.1** |

Table 4: Ablation studies of different numbers of sampled frames. We make the evaluation on three datasets in Open-ended Video-question Answering task. “Time/Epoch" denotes the time of fine-tuning for each epoch.

| Frames | MSRVTT-QA | | MSVD-QA | | TGIF | |
|---|---|---|---|---|---|---|
| | Accuracy | Time/Epoch | Accuracy | Time/Epoch | Accuracy | Time/Epoch |
| 2 | 36.1 | 740s | 34.1 | 204s | 54.3 | 245s |
| 3 | 36.7 | 897s | **34.3** | 228s | 54.7 | 310s |
| 4 | 37.0 | 1309s | **34.3** | 318s | 54.7 | 448s |
| 5 | **37.1** | 2092s | 34.0 | 377s | **55.2** | 702s |

the dataset, e.g., 1501 labels for the MSRVTT-QA.

**(2) Multiple-choice Video-question Answering.** For Multiple-choice Video-question Answering, where questions and candidate answers are sentences, we concatenate the question and answer candidates using the [SEP] token. We predict the answer by selecting the candidate with the highest output logit.

**(3) Video-text Retrieval.** For Video-text Retrieval, the retrieval process includes text-to-video and video-to-text retrieval. Each modality is extracted, followed by a comparative analysis. Predictions are generated through MLPs.

**Model Effectiveness.** Tab. 1, Tab. 6, and Tab. 3 highlight our method's superiority over baseline approaches across various datasets and evaluation criteria. Our method consistently outperforms alternatives in nearly all aspects. The conventional fine-tuning method requires approximately 897 seconds per epoch on the MSRVTT dataset when selecting three frames from the entire video (Tab. 4). In contrast, our method achieves nearly real-time communication, reducing the time delay between the cloud and the device to as little as 5.55ms, depending on internet conditions. Moreover, our method surpasses the fine-tuning approach in performance, indicating superior fast generalization on personalized samples. As shown in Tab. 2, we compare our method with the hypernetwork-based method DUET (Lv et al., 2023b), which suffers from overparameterization. Overparameterization can improve generalization but also increases the risk of overfitting, especially with noise and non-representative features (Sankararaman et al., 2020). Hypernetworks need to generate output parameters for each image frame, incurring significant cloud computing and communication costs. Our method leverages single-frame video information, achieving commendable performance while substantially reducing reasoning and training durations. Compared to the conventional HyperNetwork approach (DUET) (Lv et al., 2023b), which relies on a (5 $\times$ 192) shaped input and 28.2M FLOPs calculation, our method uses only a (1 $\times$ 192) shaped input and 0.62M FLOPs calculation, demonstrating superiority in both inference latency and accuracy, as illustrated in Table 2.

**Model Extensibility.** Our proposed approach exhibits the capacity to enhance accuracy while concurrently reducing time delays across all datasets associated with the three specific tasks. As delineated in Tab. 3, our methodology yields a significant performance improvement, particularly in the domain of text-video retrieval. Our assessment

Table 5: Time delay of our framework in different circumstances (various internet speeds) on different datasets. " ↑ " denotes the upload process from the device to cloud. " ↓ " denotes the parameters downloaded from the cloud to device.

| Datasets | Size | 4G: 5MB/s | 4G: 15MB/s | 5G: 50MB/s | 5G: 100MB/s |
|---|---|---|---|---|---|
| MSRVTT | ↑:0.75KB<br>↓:568.5KB | ↑:0.15ms<br>↓:111ms | ↑:0.05ms<br>↓:37.0ms | ↑:0.01ms<br>↓:11.1ms | ↑:0.007ms<br>↓:5.55ms |
| MSVD | ↑:0.75KB<br>↓:379.4KB | ↑:0.15ms<br>↓:74ms | ↑:0.05ms<br>↓:24.7ms | ↑:0.01ms<br>↓:7.41ms | ↑:0.007ms<br>↓:3.71ms |
| TGIF | ↑:0.75KB<br>↓:583.8KB | ↑:0.15ms<br>↓:114ms | ↑:0.05ms<br>↓:38.0ms | ↑:0.01ms<br>↓:11.4ms | ↑:0.007ms<br>↓:5.70ms |

Table 6: Results of our proposed framework in the multiple-choice Video QA task.

| Methods | MSRVTT-QA | |
|---|---|---|
| | Accuracy | Time Delay |
| F-linear | 3.58 | ≥60000ms |
| Fine-tuning | 75.6 | ≥60000ms |
| F-hyper | 46.0 | ≥5.70ms |
| Ours | **76.2** | ≥5.70ms |

encompasses a two-directional approach, addressing both text-to-video retrieval and video-to-text retrieval. Remarkably, our method surpasses the majority of baseline methods, not only in terms of accuracy but also with respect to time delays. In the context of multiple-choice Video QA, our approach consistently asserts its superiority across all evaluation metrics. This is evident from the data presented in Tab. 6.

### 4.5 Ablation Studies

**Number of Sampled Frames.** In Tab. 4, we examine how the number of sampled frames $D$ affects results and fine-tuning time per epoch in Open-ended Video-question Answering across three datasets. Using a batch size of 256 on a single A100 GPU (80G), we find that increasing frames significantly raises fine-tuning time per epoch. For instance, fine-tuning with two frames on MSRVTT-QA takes 740 seconds, while five frames take 2092 seconds. Performance peaks at three or four frames, so we set $D$ to three to balance performance and training cost. The HyperNetwork generates parameters based on the input sample distribution, with higher dimensional information helping to find more discriminative features. Increasing the number of frames improves dynamic parameter performance but also increases training time. Hence, three frames are chosen to balance training time and performance.

**Different Modules in Our Proposed Framework.** In Tab. 1, we introduce three baseline methods to evaluate our framework. We compare the fine-tuning model with our proposed model to demonstrate its superiority. The "F-linear" model is used for an ablation study of the classifiers post multi-modal feature extraction, while the "F-hyper" model examines the hyper-network and adaptive generator. Notably, omitting the adaptive generator leads to sub-optimal performance, particularly on datasets like MSRVTT-QA, MSVD-QA, and TGIF, due to over-fitting and spurious correlations between visual cues and predictions. Our adaptive generator significantly improves the hyper-network's performance by incorporating anchor-frame distribution reasoning.

**Time Delay between Cloud and Device in Various Internet Conditions.** Tab. 5 illustrates the time delay associated with data transmission between cloud services and devices under various internet conditions. These scenarios represent older mobile devices with limited or optimal 4G connectivity and modern devices with basic or advanced 5G connectivity. Our method shows significant improvements in time delay.

## 5 Conclusion

We developed an on-device multi-modal model adaptation framework, which revolutionizes on-device model adaptation for lightweight multi-modal models. Rigorous testing has demonstrated its efficacy and practicality across various scenarios and datasets.

## 6 Limitations

The StableSynthNet framework, while showing great potential, may benefit from further exploration in terms of scalability, particularly in how it adapts to an expanding range of data diversity and volume. Additionally, its cross-domain applicability could be a subject for future research to ensure that the framework remains versatile across different application scenarios. The integration process with various existing systems might offer opportunities for smoother deployment, and the long-term maintenance of the model could be optimized for evolving data landscapes. Lastly, the ethical use and transparency of the model's decision-making, as with any AI system, are important aspects that can be continually enhanced to build trust and ensure responsible AI development.

## References

Sabbir Ahmed, Abdullah Al Arafat, Mamshad Nayeem Rizve, Rahim Hossain, Zhishan Guo, and Adnan Siraj Rakin. 2023. Ssda: Secure source-free domain adaptation. In *ICCV*.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2020. Invariant risk minimization.

Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. 2019. Progressive feature alignment for unsupervised domain adaptation. In *CVPR*.

Pengfei Chen, Leida Li, Jinjian Wu, Weisheng Dong, and Guangming Shi. 2021. Unsupervised curriculum domain adaptation for no-reference video quality assessment. In *ICCV*.

Zhengyu Chen and Donglin Wang. 2021. Multi-initialization meta-learning with domain adaptation. In *ICASSP*.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. 2022. Fedavg with fine tuning: Local updates lead to representation learning. *NIPS*.

Victor G. Turrisi da Costa, Giacomo Zara, Paolo Rota, Thiago Oliveira-Santos, Nicu Sebe, Vittorio Murino, and Elisa Ricci. 2022. Dual-head contrastive domain adaptation for video action recognition. In *WACV*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Tan M. Dinh, Anh Tuan Tran, Rang Nguyen, and Binh-Son Hua. 2022. Hyperinverter: Improving stylegan inversion via hypernetwork. In *CVPR*.

David Ha, Andrew Dai, and Quoc V. Le. 2017. Hypernetworks. In *ICLR*.

A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le. 2019. Searching for mobilenetv3. In *ICCV*.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *Preprint*, arXiv:1704.04861.

Mengze Li, Ming Kong, Kun Kuang, Qiang Zhu, and Fei Wu. 2020. Multi-task attribute-fusion model for fine-grained image recognition. In *Optoelectronic Imaging and Multimedia Technology VII*, volume 11550, pages 114–123. SPIE.

Yuncheng Li, Yale Song, Liangliang Cao, Joel Tetreault, Larry Goldberg, Alejandro Jaimes, and Jiebo Luo. 2016. Tgif: A new dataset and benchmark on animated gif description. In *CVPR*.

Yawei Luo, Ping Liu, Tao Guan, Junqing Yu, and Yi Yang. 2020. Adversarial style mining for one-shot unsupervised domain adaptation. In *NIPS*.

Zheqi Lv, Zhengyu Chen, Shengyu Zhang, Kun Kuang, Wenqiao Zhang, Mengze Li, Beng Chin Ooi, and Fei Wu. 2023a. Ideal: Toward high-efficiency device-cloud collaborative and dynamic recommendation system. *arXiv preprint arXiv:2302.07335*.

Zheqi Lv, Wenqiao Zhang, Shengyu Zhang, Kun Kuang, Feng Wang, Yongwei Wang, Zhengyu Chen, Tao Shen, Hongxia Yang, Beng Chin Ooi, et al. 2023b. Duet: A tuning-free device-cloud collaborative parameters generation framework for efficient device model generalization. In *WWW*, pages 3077–3085.

Ahmadreza Mosallanezhad, Mansooreh Karami, Kai Shu, Michelle V Mancenido, and Huan Liu. 2022. Domain adaptive fake news detection via reinforcement learning. In *WWW*, pages 3632–3640.

Subhankar Roy, Evgeny Krivosheev, Zhun Zhong, Nicu Sebe, and Elisa Ricci. 2021. Curriculum graph co-teaching for multi-target domain adaptation. In *CVPR*.

M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*.

Karthik A. Sankararaman, Soham De, Zheng Xu, W. Ronny Huang, and Tom Goldstein. 2020. The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In *ICML*. JMLR.org.

Yang Shu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. 2019. Transferable curriculum for weakly-supervised domain adaptation. In *AAAI*.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. 2021. Training data-efficient image transformers amp & distillation through attention. PMLR.

Alex Jinpeng Wang, Yixiao Ge, Rui Yan, Ge Yuying, Xudong Lin, Guanyu Cai, Jianping Wu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. 2023a. All in one: Exploring unified video-language pre-training. *CVPR*.

Haixin Wang, Jinan Sun, Xiang Wei, Shikun Zhang, Chong Chen, Xian-Sheng Hua, and Xiao Luo. 2023b. Dance: Learning a domain adaptive framework for deep hashing. In *WWW*, pages 3319–3330.

Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. 2017. Video question answering via gradually refined attention over appearance and motion. ACM MM.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. CVPR.

Yikai Yan, Chaoyue Niu, Renjie Gu, Fan Wu, Shaojie Tang, Lifeng Hua, Chengfei Lyu, and Guihai Chen. 2022. On-device learning for model personalization with large-scale cloud-coordinated domain adaption. In *KDD*, pages 2180–2190.

Jiangchao Yao, Feng Wang, Kunyang Jia, Bo Han, Jingren Zhou, and Hongxia Yang. 2021. Device-cloud collaborative learning for recommendation. In *KDD*.

Liping Yi, Xiaorong Shi, Nan Wang, Ziyue Xu, Gang Wang, and Xiaoguang Liu. 2023. pfedlhns: Personalized federated learning via local hypernetworks. In *ICANN 2023*.

Junkun Yuan, Xu Ma, Defang Chen, Kun Kuang, Fei Wu, and Lanfen Lin. 2022a. Domain-specific bias filtering for single labeled domain generalization. *International Journal of Computer Vision*, pages 1–20.

Junkun Yuan, Xu Ma, Defang Chen, Kun Kuang, Fei Wu, and Lanfen Lin. 2022b. Label-efficient domain generalization via collaborative exploration and generalization. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 2361–2370.

Chris Zhang, Mengye Ren, and Raquel Urtasun. 2020. Graph hypernetworks for neural architecture search. In *ICLR*.

Fengda Zhang, Kun Kuang, Yuxuan Liu, Long Chen, Chao Wu, Fei Wu, Jiaxun Lu, Yunfeng Shao, and Jun Xiao. 2021. Unified group fairness on federated learning. *arXiv preprint arXiv:2111.04986*.

Ruohan Zhang, Tianzi Zang, Yanmin Zhu, Chunyang Wang, Ke Wang, and Jiadi Yu. 2023a. Disentangled contrastive learning for cross-domain recommendation. In *International Conference on Database Systems for Advanced Applications*. Springer.

Shengyu Zhang, Xusheng Feng, Wenyan Fan, Wenjing Fang, Fuli Feng, Wei Ji, Shuo Li, Wang Li, Shanshan Zhao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2023b. Video audio domain generalization via confounder disentanglement. In *AAAI*.

Yang Zhang, Philip David, Hassan Foroosh, and Boqing Gong. 2019. A curriculum domain adaptation approach to the semantic segmentation of urban scenes. *TPAMI*, 42(8):1823–1841.