electronics

MDPI

Article

Privacy-Preserving Real-Time Action Detection in Intelligent Vehicles Using Federated Learning-Based Temporal Recurrent Network †

Alpaslan Gökcen [1,2,*] and Ali Boyacı [1,3,‡]

[1] Computer Engineering Department, İstanbul Commerce University, Istanbul 34840, Turkey; aboyaci@ticaret.edu.tr
[2] Turkcell İletişim Hizmetleri, Maltepe, Istanbul 34854, Turkey
[3] Grid Communications and Security Group, Electrification and Energy Infrastructures Division, Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA
* Correspondence: alpaslan.gokcen@turkcell.com.tr
† This manuscript has been authored in part by UT–Battelle, LLC, under contract DE–AC05–00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).
‡ Ali Boyaci was with İstanbul Commerce University and now with Oak Ridge National Lab.

Abstract: This study introduces a privacy-preserving approach for the real-time action detection in intelligent vehicles using a federated learning (FL)-based temporal recurrent network (TRN). This approach enables edge devices to independently train models, enhancing data privacy and scalability by eliminating central data consolidation. Our FL-based TRN effectively captures temporal dependencies, anticipating future actions with high precision. Extensive testing on the Honda HDD and TVSeries datasets demonstrated robust performance in centralized and decentralized settings, with competitive mean average precision (mAP) scores. The experimental results highlighted that our FL-based TRN achieved an mAP of 40.0% in decentralized settings, closely matching the 40.1% in centralized configurations. Notably, the model excelled in detecting complex driving maneuvers, with mAPs of 80.7% for intersection passing and 78.1% for right turns. These outcomes affirm the model's accuracy in action localization and identification. The system showed significant scalability and adaptability, maintaining robust performance across increased client device counts. The integration of a temporal decoder enabled predictions of future actions up to 2 s ahead, enhancing the responsiveness. Our research advances intelligent vehicle technology, promoting safety and efficiency while maintaining strict privacy standards.

Keywords: data privacy; real-time action detection; intelligent vehicles; federated learning

Citation: Gökcen, A.; Boyacı, A. Privacy-Preserving Real-Time Action Detection in Intelligent Vehicles Using Federated Learning-Based Temporal Recurrent Network. *Electronics* **2024**, *13*, 2820. https://doi.org/10.3390/electronics13142820

Academic Editor: José Carlos Castillo

Received: 14 May 2024
Revised: 5 June 2024
Accepted: 5 June 2024
Published: 18 July 2024

1. Introduction

In a rapidly evolving digital landscape, the demand for real-time action detection has become increasingly critical across various domains, including surveillance, autonomous vehicles, and human–computer interaction. Online action detection, the process of identifying and localizing actions in streaming video data, is pivotal in enabling responsive and adaptive systems. However, this task is challenging, particularly in terms of privacy preservation and scalability.

The traditional approaches to online action detection often involve centralized data collection and model training [1–10]. While effective, these methods raise concerns about data privacy and communication efficiency as they require the consolidation of sensitive

information in a central repository [11]. Moreover, as the scale of data generation grows, centralized solutions face limitations in scalability [12].

To address these challenges, we propose an innovative approach that combines federated learning (FL) with a temporal recurrent network (TRN) [3] tailored for online action detection. FL enables distributed edge devices to collaboratively train action detection models while preserving the confidentiality of their data [13]. This approach enhances data privacy and alleviates scalability issues by enabling edge devices to independently train their models on localized data sources, such as driving videos and bus sensor information [14].

Our federated TRN model is designed to capture the temporal dependencies inherent in video data, making it well-suited for real-time action detection. It processes sequential frames from video streams, analyzing how actions evolve over time. A critical feature of our TRN is its ability to anticipate future actions, effectively predicting actions up to 2 s into the future. This anticipation is achieved through a temporal decoder that learns feature representations and integrates the predicted future actions with historical evidence.

This paper presents the architecture and methodology of our FL-based TRN for online action detection. We provide the experimental results, demonstrating the competitive performance of our model on the Honda HDD dataset [15] and TVSeries dataset in both centralized and decentralized settings. Moreover, we discuss the advantages of our approach, including data privacy, scalability, and adaptability. Therefore, in this study, the contributions are threefold. Firstly, we introduce a novel FL-based TRN for online action detection, showcasing its efficacy in capturing temporal dependencies and predicting future actions. Secondly, our proposed approach achieves a competitive performance, as evidenced by the impressive mean average precision (mAP) scores on the Honda HDD dataset. Lastly, our methodology emphasizes data privacy, scalability, and adaptability, ensuring its applicability in diverse real-world settings.

Our privacy-preserving model enables real-time action detection in intelligent vehicles without sharing raw data, safeguarding user privacy. The model enhances the performance and robustness by integrating federated learning with temporal recurrent networks, ensuring better generalization across diverse driving environments. This approach significantly advances intelligent vehicle technology, promoting safety and efficiency while maintaining privacy standards.

In our research endeavor, we have employed widely used algorithms, namely the Federated Mean, Federated Median, and Federated Proxy, formally referred to as the Federated Proximity Optimization Algorithm.

The rest of this paper is organized as follows: Section 2 reviews the related work in the fields of online action detection and federated learning. Section 3 provides a detailed description of our methodology, including the architecture of the TRN and the integration of FL. Section 4 outlines our experimental setup, including the datasets and evaluation metrics. Section 5 presents our experimental results. Section 6 includes a discussion and conclusion. Finally, Section 7 offers future directions, highlights the significance of our FL-based approach in the context of online action detection, and explores potential avenues for future research.

2. Related Work

In this section, our objective is to furnish a comprehensive survey and synthesis of the current state of the scholarly investigations in terms of online action detection and federated learning. By meticulously examining the extant literature, we aim to encapsulate the diverse spectrum of research contributions, methodologies, and advancements within these fields. This overview offers readers a nuanced understanding of the existing knowledge landscape, setting the stage for the subsequent in-depth discussions and analyses within the broader discourse.

2.1. Online Action Detection

Given a real-time video stream encompassing one or more actions, we aim to discern the actions of interest manifesting in each video frame. Diverging from the conventional approaches and assuming the simultaneous availability of the entire video sequence, the online action detection problem necessitates the immediate processing of each frame upon its arrival, devoid of access to future information. Formally, we aim to infer, for each frame I_t within an image sequence, a probability distribution $p_t = [p_t^0, p_t^1, p_t^2, \ldots, p_t^K]$ spanning K potential actions. This estimation relies solely on the information from past and current frames, denoted as $f(I_1, I_2, \ldots, I_t)$, wherein p_0^t signifies the probability of the "background" scenario where no action is transpiring.

Online action detection is a rapidly evolving field focusing on real-time action recognition and localization in video streams. The traditional methods in this domain often rely on handcrafted features and models that process each frame independently, which may limit their ability to capture temporal dependencies. Recent advancements have witnessed the adoption of deep learning techniques, including Convolutional Neural Networks (CNNs) [16] and Recurrent Neural Networks (RNNs) [17], to improve the action detection accuracy.

The notable approaches include the use of two-stream networks that combine spatial and temporal information as well as Temporal Convolutional Networks (TCNs) designed to capture long-range dependencies in video data [18]. These methods have demonstrated impressive results in action recognition tasks.

A substantial amount of studies in the literature are devoted to action and activity recognition in various types and uses of videos. Consumer videos [19], surveillance videos [20], and first-person videos taken using body-worn cameras [21–23] are examples. The previous works in the literature are based on handcrafted visual characteristics like HOG [24], HOF [24], and MBH [25], as well as motion features like enhanced dense trajectories [26]. Deep convolutional networks have mostly been used in the recent approaches. Simonyan and Zisserman [27], for example, suggested a two-stream convolutional network that accepts RGB frames as well as optical streams as input [28]. Others, such as Tran et al. [29] and Carreira et al. [30], eschew precomputing the optical flow and instead use 3D convolution to learn end-to-end time information. To record time and motion information [31], Recurrent Neural Networks (RNN) such as long short-term memory (LSTM) [32] and gated recurrent units (GRUs) [33] have frequently been used [34]. However, most of these methods are designed for cut-and-paste videos and may not be immediately relevant to long video sequences with multiple actions and backgrounds.

2.2. Federated Learning in Computer Vision

Federated learning (FL) has gained prominence in various fields, including computer vision, for its privacy-preserving ability and distributed model training. In the context of computer vision, FL has been applied to object detection, image classification, and video analysis.

Recent studies have explored FL-based techniques for collaborative model training across distributed cameras in surveillance systems, preserving data privacy while improving object recognition [35,36]. Additionally, FL has been employed in multimodal settings, where sensors from different devices contribute to a unified model, enhancing the accuracy of action recognition [37].

The conceptualization of a driver activity recognition (DAR) model incorporating a federated learning (FL) approach has been introduced. The inherent complexity of driver activity recognition mandates nuanced differentiation among various actions, compelling the utilization of diverse training data from multiple sources. The innovative FL-based DAR model addresses the imperatives of data privacy preservation and the security concerns often associated with centralized configurations and establishes its efficacy by showcasing competitive performance in centralized and decentralized settings, as substantiated in [38].

Accurate and Privacy-Preserving Person Localization Using Federated Learning and the Camera Surveillance Systems of Public Places is proposed [39]. In contrast to the

prevailing methodologies reliant on the Euclidean distance measurements of embedding vectors storing facial features for person location, the proposed approach employs a more accurate strategy by training a machine learning model. Additionally, a federated learning technique is employed to mitigate the potential of compromising sensitive information associated with sharing images of visitors in public places for model training. This ensures the computation of the model in a privacy-preserving manner.

The Federated Few-Shot Learning framework, FedFSLAR, was designed to enhance video-based action recognition by combining federated learning with few-shot learning. This approach addresses the challenges of limited annotated data and model biases in FL clients, leveraging 3D CNNs within a meta-learning paradigm to capture the temporal correlations in video frames more effectively. The experimental results on benchmark datasets demonstrate that FedFSLAR significantly improves the action recognition performance under non-IID data conditions, showcasing the potential of integrating pre-trained feature backbones with federated learning for few-shot learning tasks [40].

The Federated Skeleton-based Action Recognition (FSAR) paradigm serves to address the privacy concerns and training instability in skeleton-based action recognition. By incorporating an Adaptive Topology Structure (ATS) and Multi-grain Knowledge Distillation (MKD), FSAR effectively separates generalization and personalization, enabling stable and privacy-preserving model training across heterogeneous human topology graphs. Extensive experiments show that FSAR outperforms the existing FL-based methods, achieving comparable performance while protecting user privacy [41].

2.3. Challenges and Opportunities

While the existing research has made significant strides in online action detection and federated learning, there remain several challenges and opportunities. The privacy preservation in FL remains a critical concern, particularly in scenarios involving video data from edge devices. Moreover, achieving efficient model updates and communication in FL for real-time applications is an ongoing research area.

Given the constrained computational resources inherent to vehicular edge devices, efficient model development is imperative. Federated learning (FL) facilitates collaborative model development among data parties to address this challenge. FL operates by preserving data privacy and minimizing the communication requirements, presenting a viable strategy to surmount the computational constraints in the context of vehicular edge devices.

Our work addresses these challenges by proposing a novel FL-based temporal recurrent network (TRN) for online action detection. By adapting TRN for FL and integrating predictive capabilities, we provide a versatile and privacy-conscious solution for real-time action detection across distributed edge devices.

3. Methodology

Within this section, we present an elaborate elucidation of our chosen methodology, directing particular attention to the instantiation of the federated learning (FL)-based temporal recurrent network (TRN) for the purpose of online action detection. We intend to provide readers with a comprehensive understanding of the intricacies involved in implementing and integrating FL principles within the temporal context, as encapsulated by the TRN framework. This entails a detailed exploration of the components, processes, and rationale underlying our chosen approach, thereby fostering clarity and insight into the methodological underpinnings of our study.

3.1. Architecture of FL-Based TRN

Our approach leverages FL to collaboratively train action detection models across distributed edge devices while preserving data privacy. The architecture of our FL-based TRN consists of the following key components:

Temporal Recurrent Network (TRN): TRN is designed to capture temporal dependencies within video data, as shown in Figure 1. It recurrently processes sequential frames,

enabling it to analyze how actions evolve over time. This is achieved through recurrent layers and long short-term memory (LSTM) [32], which maintains past frames' memory and features.

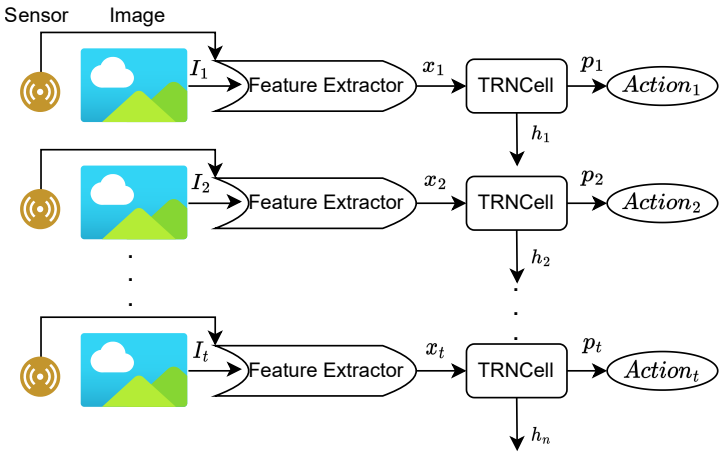

Figure 1. Temporal recurrent network.

Our TRN network architecture incorporates specialized cells featuring a temporal decoder, a future gate, and a spatiotemporal accumulator (STA), as illustrated in Figure 2. LSTM was used as a backbone of STA and the temporal decoder. The temporal decoder is designed to acquire a feature representation and forecast forthcoming actions within a given sequence. The future gate functions by accepting a set of concealed states from the decoder and incorporates these attributes into the future context, as shown in Algorithm 1. The STA mechanism plays a crucial role in this context as it assimilates spatiotemporal features derived from historical, present, and anticipated future data. The STA model is proficient in assessing the action taking place in the present frame by integrating information across various temporal dimensions.

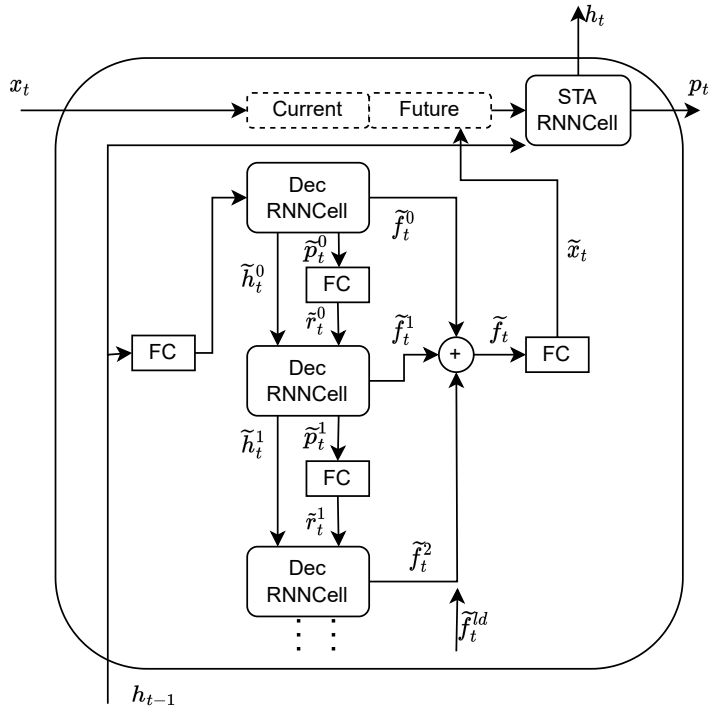

Figure 2. The TRN cell.

The TRN model functions sequentially to anticipate forthcoming actions and their associated hidden states across a defined number of time steps ($\widetilde{l_d}$). The evolving states of

the decoder at each time step after t are denoted as $\widetilde{h}_t^0, \widetilde{h}_t^1, \ldots, \widetilde{h}_t^{l_d}$. Initialization involves an all-zero input at the initial time step. Subsequently, the decoder incorporates predicted action scores ($\widetilde{r}_t^{i-1}$) in later time steps, embedding these scores using a linear transformer.

Algorithm 1: Workflow of a TRN Cell

Input: Image feature x_t and previous hidden state h_{t-1}
Output: Probabilities p_t and current hidden state h_t

Initialization: $\widetilde{h}_t^{-1} \leftarrow h_{t-1}$ embedded by an FC layer;
$\widetilde{r}_t^{-1} \leftarrow$ all zeros;
for $i = 0$ **to** l_d **do**
    Update $\widetilde{h}_i^t$ using $\widetilde{r}_t^{i-1}$ and $\widetilde{h}_t^{i-1}$;
    Compute f_t^i and $\widetilde{p}_t^i$ using $\widetilde{h}_t^i$;
    Update $\widetilde{r}_t^i$ using $\widetilde{p}_t^i$;
end
Compute future context features $\widetilde{x}_t$ using Equation (1);
Update h_t with $STA(h_{t-1}, [x_t, \widetilde{x}_t])$;
Compute p_t using Equation (2);

The future gate module captures the feature representation of forthcoming context by extracting hidden states from the decoder and performing a computational operation, typically involving an average pooling operator followed by a fully connected (FC) layer. More specifically, the feature representing future context ($\widetilde{x}_t$) is computed through the averaging and embedding of the hidden state vector ($\widetilde{h}_t$) derived from all decoder steps. This process ensures the nuanced integration of future contextual information within the model architecture.

$$\widetilde{x}_t = \mathrm{ReLU}\left(W_f^T \mathrm{AvgPool}(\widetilde{h}_t) + b_f\right) \tag{1}$$

The spatiotemporal accumulator (STA) amalgamates the preceding hidden state (h_{t-1}), the image feature (x_t), and the forecasted future feature ($\widetilde{x}_t$). It adapts its hidden states (h_t) and calculates a distribution across potential actions using a softmax activation function:

$$p_t = \mathrm{softmax}\left(W_c^T h_t + b_c\right) \tag{2}$$

Here, W_c and b_c are parameters of the FC layer used for action classification.

The TRN model generates predictions not solely for the present frame (t) but extends its forecasting horizon to encompass the subsequent d time steps. In the pursuit of concurrently optimizing online action detection and prediction tasks, the model integrates the accumulator and decoder losses throughout the training process. This concerted approach during training facilitates the amalgamation of detection and prediction objectives, contributing to the holistic enhancement of the model's performance. The loss function for one input sequence is a combination of the loss for the predicted actions at the current time step and the losses for the predicted actions at the next d time steps:

$$\sum_t (\mathrm{loss}(p_t, l_t) + \alpha \sum_{i=0}^{l_d} \mathrm{loss}(\widetilde{p}_t^i, l_{t+i})) \tag{3}$$

At this specific juncture, $\widetilde{p}_t^i$ signifies the predicted action probabilities generated by the decoder for step i subsequent to time t, while l_t denotes the actual ground truth. The expression loss designates the cross-entropy loss, and α serves as a scaling factor. The network undergoes optimization through offline training, leveraging labels for both the present and future frames. In the testing phase, the model utilizes forecasted future information without direct access to the actual future frames, characterizing it as an online

model. The optimization objective minimizes the discrepancy between predicted and actual action probabilities, as governed by the cross-entropy loss term.

Integration of FL: FL is seamlessly integrated into the TRN model. Each edge device with video data locally trains its TRN model, capturing temporal patterns and actions. Periodically, a central server aggregates these local model updates, creating a global TRN model that benefits from the collective knowledge of all devices, as shown in Figure 3. This decentralized approach preserves data privacy and enhances model accuracy.

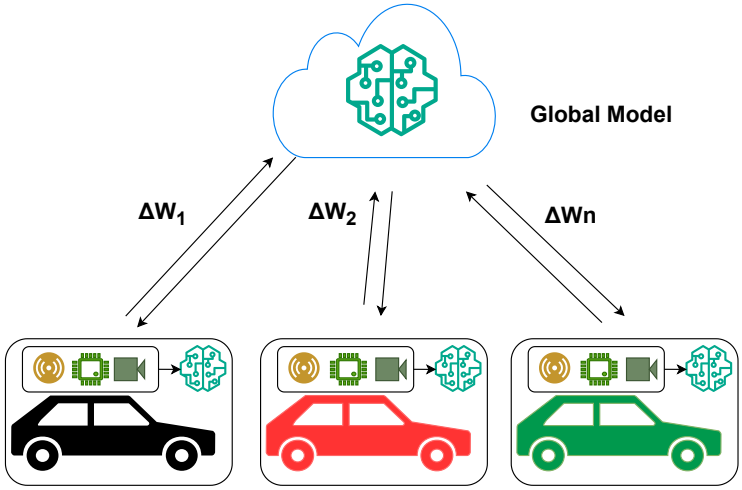

Figure 3. Federated learning on intelligent vehicles.

3.2. Data Preprocessing and Model Training

We perform data preprocessing steps to prepare the data for training, including frame extraction, resizing, and feature extraction. We extract relevant features from each frame to feed into the TRN model.

Three different federated learning types have been implemented, namely Federated Mean Aggregation, Federated Median Aggregation, as shown in Algorithm 2, and FedProx [42], illustrated in Algorithm 3. We have implemented the identical algorithm for Federated Mean Aggregation, with the only distinction lying in the aggregation method, where the mean is employed as the primary metric.

Algorithm 2: Federated Learning with Median-based Aggregation

Input: Server, Client devices $C_1, C_2, \ldots, C_N$
Output: Improved global model M'
Initialization: Initialize M with random weights on the server;
for *each communication round* $t = 1, 2, \ldots, T$ **do**
  foreach *client* C_i **do**
    Client C_i:
    Sample a random subset of local data D_i;
    Compute the local update ΔM_i on C_i using D_i and M;
    `// Which means, Each Client (`C_i`) uses Global Model(`M`) weights and local data(`D_i`) for training itself.`
    ; Share ΔM_i with the server;
  end
  Server:
  Aggregate client updates $M' \leftarrow \text{Median}(\Delta M_1, \Delta M_2, \ldots, \Delta M_N)$;
  Update the global model: $M \leftarrow M'$;
end

Algorithm 3: FedProx Algorithm

Input: Central model, Clients
Output: Updated central model
Initialize central model;
Distribute the model to all clients;
for *each iteration* **do**
    Collect updated models from all clients;
    Update the central model using FedProx;
    Distribute the updated central model to all clients;
end

FedProx Model Update:

Input: Central model, Local models from clients
Output: Updated central model
Initialize *global_weights* as zeros;
for *each client* **do**
    Train local model with local data;
    Get updated weights from local model;
    Add local weights to *global_weights*;
end
Average *global_weights*;
Calculate proximal term Δ as
    Δ = central model − global_weights;
    Update central model using FedProx as
        central model = central model − $\alpha \cdot \Delta$;

The data were distributed randomly among client devices. The subsequent results section will include detailed model parameters and a discussion of outcomes.

For the model's training, each edge device independently conducts training on its localized data using the temporal recurrent network (TRN) model. This localized dataset comprises sequences of video frames annotated with corresponding action labels. The training process encompasses optimizing model parameters, including those of the temporal decoder, employing the Adam optimizer.

3.3. Future Action Performance

We further assess the capacity for action anticipation, forecasting the subsequent action in a given sequence. Our findings indicate comparable performance in terms of *mAP* compared to the existing state-of-the-art methods in this context, notwithstanding that anticipation does not constitute the primary emphasis of our current investigation.

Our FL-based TRN excels in anticipating future actions, a critical capability for real-time action detection. By considering historical patterns and predicted future actions, the model enhances its accuracy in recognizing actions as they unfold. The temporal decoder and spatiotemporal accumulator (STA) play a crucial role in this anticipation, extracting spatiotemporal features and integrating future predictions seamlessly into the detection process.

This architecture and methodology enable our FL-based TRN to excel in online action detection while addressing privacy concerns and ensuring scalability.

4. Experimental Setup

Proceeding to discuss the experimental setup, we present a comprehensive depiction of the experimental configuration employed to assess the efficacy and performance of our temporal recurrent network (TRN) for online action detection. This encompasses a detailed delineation of the configuration, including the hardware and software components, the intricacies of the dataset employed for training and evaluation, as well as the specific parameters and methodologies utilized throughout the experimentation process. The

objective is to offer the readers a thorough understanding of the conditions under which the federated learning (FL)-based TRN model is evaluated, thereby enhancing transparency and facilitating the interpretation of the subsequent results and analyses.

4.1. Datasets

We conducted our experiments using the Honda HDD and TVSeries datasets, as shown in Figures 4 and 5, widely recognized benchmark datasets for action detection. The datasets include a diverse range of action classes and real-world scenarios, making them suitable for evaluating the robustness and accuracy of our FL-based approach.

Figure 4. Sample images from Honda dataset.

The Honda HDD dataset contains video sequences captured from multiple cameras in various environments, providing challenges such as different lighting conditions, occlusions, and complex background scenes. We used a subset of the dataset for training and testing, ensuring a balanced distribution of action classes. The datasets are distributed across the clients to ensure that each client receives an equal number of action events for each action. In federated learning, the use of independently and identically distributed (IID) data across clients ensures uniformity in training, leading to more consistent and reliable model performance. When data are non-IID, clients possess data with varying distributions, resulting in significant disparities in local updates. This lack of homogeneity in data distribution causes the aggregated global model to perform poorly as it struggles to generalize across the diverse local data environments [43].

The HDD dataset comprises 137 driving sessions conducted in the San Francisco Bay Area. It includes frame-level annotations for 11 distinct goal-oriented actions, along with non-visual sensor readings.

Figure 5. Sample images from TvSeries dataset.

The dataset was used for training and testing. We used 100 sessions for training and and 37 sessions for testing, which is consistent with previous works [3,15]. Random splitting of the dataset disrupts the temporal flow. For this reason, datasets are compiled on a session basis.

The TVSeries dataset represents a substantial and authentic collection intended for action detection tasks, characterized by its large scale and realism. Comprising 16 h of footage extracted from six contemporary television series, it offers a diverse and comprehensive corpus for analysis. The dataset encompasses thirty distinct action classes, each meticulously annotated with precise temporal markers indicating the commencement and cessation of the actions. Furthermore, accompanying metadata enrich the dataset by furnishing supplementary information on various contextual factors, including instances featuring a solitary individual, occurrences of occlusion, or partial coverage of the action. Such detailed annotations enable researchers to scrutinize methodological performance across a spectrum of challenging scenarios, thereby enhancing the evaluative capabilities of action detection methodologies [7].

4.2. Training and Testing Protocols

To evaluate the performance of our FL-based TRN, we employed a standard training and testing protocol. The dataset was divided into training and testing sets, with a

73/27 split. This enabled us to assess the model's ability to generalize to unseen data while ensuring sufficient training samples.

Each edge device independently trained its TRN model using its localized training data for training. Training iterations and batch sizes were consistent across all devices to maintain fairness in model convergence. The central server aggregated local model updates after a predefined number of training iterations.

The testing phase involved evaluating the models on the held-out testing data. We measured performance metrics, including mean average precision (mAP), to assess the accuracy of action detection and localization. Additionally, we recorded execution times to evaluate the real-time capabilities of our FL-based TRN.

4.3. Hardware and Software Configuration

The experiments were conducted on a single computer equipped with a dedicated GPU to accelerate model training and inference. The computer specs are an i5 13400F CPU, NVIDIA 3060Ti GPU, and 32 GB System Memory. We used Python (v3.9) and popular deep learning libraries, including TensorFlow (v2.10) and PyTorch (v2.0), for model development and implementation. Federated learning was facilitated using the numpy library and PyTorch for secure and privacy-preserving model updates.

InceptionResnet-V2 [44] pre-trained on ImageNet [45] is used for extracting features of each frame.

The computer is used to simulate a central server and clients. The central server responsible for model aggregation has ample storage and computational resources to efficiently handle the incoming updates from edge devices.

4.4. Metrics

To evaluate the performance of our FL-based TRN, we employed the following metrics:

Mean Average Precision (mAP): This metric assesses the accuracy of action detection and localization. It provides a measure of how well our model identifies actions in video sequences.

The mean average precision (mAP) is calculated as

$$\text{mAP} = \frac{1}{N} \sum_{k=1}^{N} \text{AP}_k \tag{4}$$

where

N: The total number of classes or categories;

AP_k: Average Precision for class or category k.

Execution Time: We recorded the time taken for our FL-based TRN to process video frames and generate action predictions in real-time scenarios. This metric quantifies the model's efficiency and suitability for online applications.

4.5. Experimental Design

To extract features, firstly, frames were extracted from the videos at a rate of 3 frames per second. This frame rate was chosen to balance computational efficiency with capturing important visual information. Each extracted frame went through a series of preprocessing steps to ensure consistency and improve the performance of the pre-trained InceptionResnet-V2 model. Preprocessing steps involved resizing each frame to a fixed resolution of 299 $\times$ 299 pixels, which was the input size required by the InceptionResnet-V2 model. Next, the pre-processed frames were fed into the pre-trained InceptionResnet-V2 model to extract high-level feature representations. Details of these features and the specific layers from which they were extracted are described below.

Utilizing the identical configuration as detailed in Ref. [3,15], we initiated the process by sampling video frames and values from Controller Area Network (CAN) bus sensors at a rate of 3 frames per second (fps). The Conv2d_7b_1$\times$1 layer within the InceptionResnet-V2 [44] architecture, pre-trained on ImageNet [45], yields the visual features for each frame.

An additional 1 × 1 convolutional operation was applied to retain spatial information, reducing the frame features from 8 × 8 × 1536 to 8 × 8 × 20 and flattening them into 1200-dimensional vectors. Concurrently, raw sensor numeric representation underwent processing through a fully connected layer, producing 20-dimensional outputs. The visual and sensor features obtained were concatenated to create a multimodal representation for every video frame. Adhering to the methodology outlined in [15], we designated the input sequence length (l) as 90. The number of decoder steps (l_d) is regarded as a hyperparameter, subject to cross-validation in experimentation, whose (l_d) value is evaluated in Figure 6. The temporal decoder and the Spatiotemporal Accumulator (STA) employ 2000 dimensions for their hidden units. These are proposed in [15]; also, evaluation results are shown in Figure 7. The constant α was selected at a value of 1.0 in Equation (3).

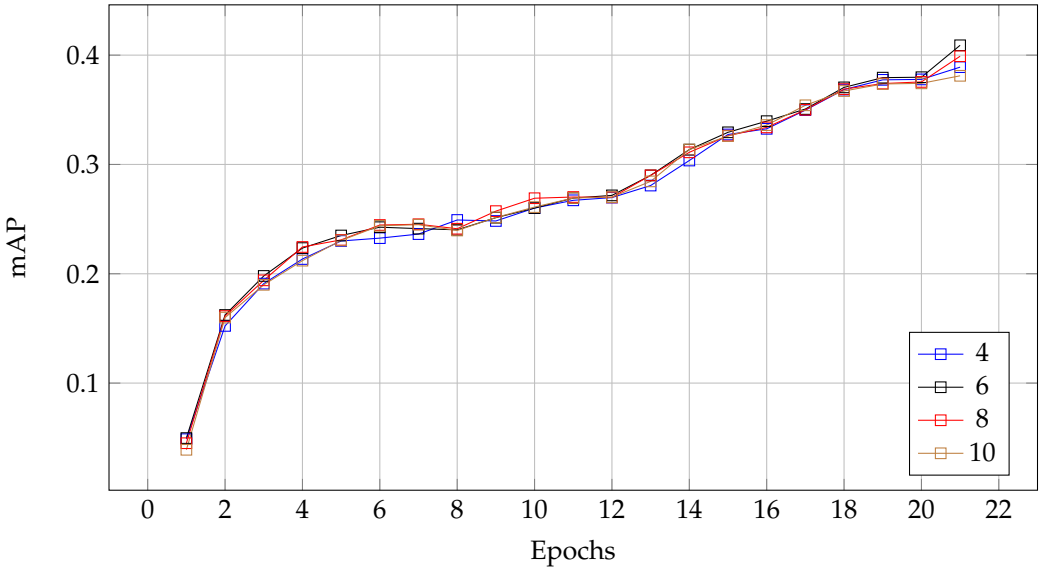

Figure 6. Selecting the l_d in terms of best mAP using HDD dataset.

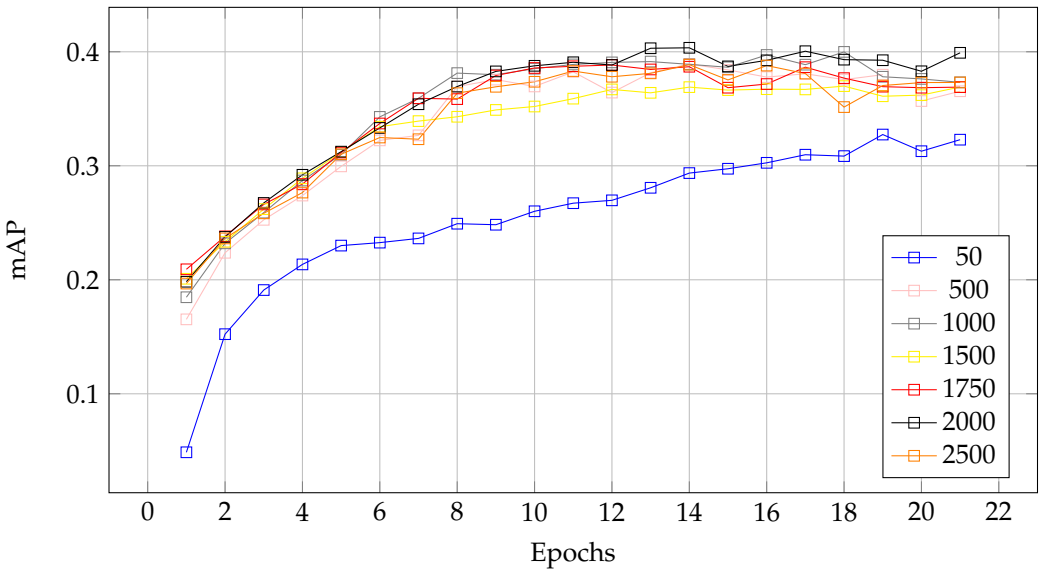

Figure 7. Selecting the hidden units of the temporal decoder size in terms of best mAP using HDD dataset.

For TVseries Dataset: In alignment with established methodology as outlined in prior literature [15], the configuration parameters employed for experimentation adhere to the standardized settings utilized in the analysis of the TVSeries dataset. Videos are sampled at a frame rate of 24 frames per second, with each video chunk comprising a

duration of 6 s. Decision-making processes are conducted at the level of video chunks, thereby facilitating performance evaluation at intervals of 0:25 s. Feature extraction utilizes two distinct methodologies: VGG-16 and two-stream (TS) CNN. Specifically, VGG-16 features are extracted from the fc6 layer of the central frame within each chunk, while two-stream features integrate appearance features derived from the Flatten 673 layer of ResNet-200 and motion features extracted from the global pool layer of BN-Inception. The concatenation of these appearance and motion features yields the two-stream feature representation. l_e is the input sequence length constrained to 64. Consistent with previous research benchmarks [15], the number of decoder steps (l_d) is fixed at 8, corresponding to a temporal span of 2 s. The hidden units of both the temporal decoder and the Spatiotemporal Attention (STA) mechanism are configured to possess dimensions of 4096, maintaining fidelity to established conventions within the field.

The selection of parameters outlined herein is grounded in meticulously considering prior research endeavors that have demonstrated noteworthy performance [3,15]. The deliberate choice of these parameters is motivated by achieving comparability with established benchmarks set by exemplary works in the field. By aligning our experimental setup with configurations proven to yield favorable outcomes in previous studies, we aim to facilitate a meaningful and rigorous comparison of our results with those reported in the existing literature. This strategic alignment ensures our findings' contextual relevance and contributes to our experimental framework's robustness.

The network weights were learned using the Adam optimizer, and data augmentation involved randomly chopping off frames and dividing the video into non-overlapping training samples. Model parameters were selected as mentioned via TRN implementation [3] with a learning rate 0.0005, l_d 2 s (6 frames in HDD dataset), weight decay 0.0005, and batch size 32 for decentralized learning approaches. A value of 8 was used for federated learning approaches, and the epoch value was 21.

The number of clients was selected as 4 due to the lack of data and computing power. The proposed FL model was evaluated with different client sizes from 2 to 4. The central model updated its weights in each epoch.

5. Results

Examining the results of our experiments aimed at appraising the efficiency of our federated learning (FL)-based temporal recurrent network (TRN) for online action detection, we provide an in-depth exploration of the results we obtained.

5.1. Performance Metrics

We employed a set of performance metrics to assess the effectiveness of our FL-based TRN in action detection:

- Mean Average Precision (mAP): This metric quantifies the accuracy of the action detection and localization. It measures the precision and recall of detected actions and comprehensively assesses our model's performance.
- Execution Time: We recorded the time taken by our FL-based TRN to process video frames and generate action predictions. This metric is crucial for evaluating the model's efficiency in real-time scenarios.
  The FL-based median aggregation and mean training time is approximately 5 h on average for each client when the client size is 4. The central model training time is 20 h.

5.2. Experimental Results

Our FL-based TRN achieved competitive results across a range of experiments, showcasing its effectiveness in online action detection:

1. We tested the l_d parameter, where the performance of action anticipation is averaged over the decoder steps. The results indicate that having a larger number of decoder steps does not necessarily guarantee better performance. Although the higher l_d

values (8 and 10) initially show a slight improvement in mean average precision (mAP), the differences diminish as the number of epochs increases. This phenomenon can be attributed to the fact that the anticipation accuracy typically decreases for longer future sequences, leading to increased noise in the input features of the Spatiotemporal Attention (STA) mechanism. Thus, while varying the number of decoder steps is crucial for optimizing the performance, the results suggest that a balanced approach is necessary to mitigate the introduction of noise.

2. mAP Scores: In both centralized and decentralized settings, our FL-based TRN consistently demonstrated competitive mAP scores. These scores reflect the model's accuracy in localizing and identifying actions within video sequences. Specifically, we achieved an mAP of 40.1 in the centralized setting and 40.0 in the decentralized setting. The model training performance with HDD and TVSeries datasets are given in Figure 8 and Figure 9 respectively.
3. Our methodology enables the model to anticipate actions for up to 2 s into the future, contributing to the system's responsiveness and adaptability. The FL-TRN model achieves comparable results with the central TRN model. Their mAP scores are 31.8 and 32.2 when the decoder step (l_d) is 6.
4. Scalability: Our approach exhibited excellent scalability as the number of client sizes increased. The model's performance remained consistent, highlighting its ability to handle large-scale deployments effectively.
5. Privacy Preservation: Our FL-based approach ensured data privacy by enabling edge devices to retain their data locally during training. This privacy-preserving mechanism is a crucial benchmark as it addresses the concerns related to sensitive video data.
6. Adaptability: The proposed approach demonstrated adaptability to dynamic environments. The model continually updated its knowledge from various edge devices, enabling it to adapt to changing action patterns and scenarios. This adaptability is a valuable feature for real-world applications.
7. Execution Time: Our FL-based TRN exhibited efficient real-time processing capabilities. It processed video frames and generated action predictions within milliseconds, making it suitable for online applications where timely responses are crucial.

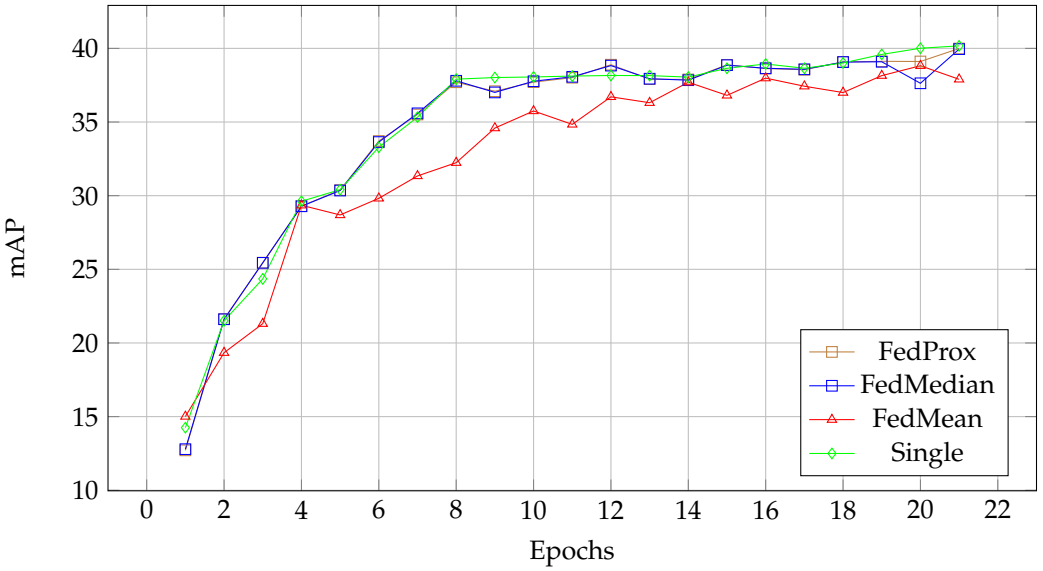

Figure 8. Methods' performance in each epoch using HDD dataset.

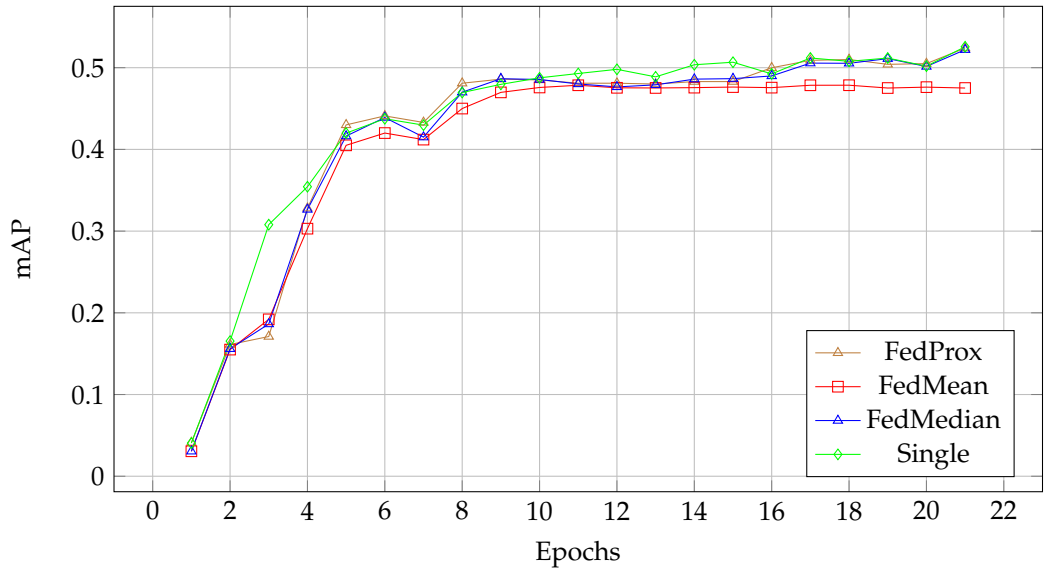

Figure 9. Methods' performance in each epoch using TVSeries dataset.

5.3. Comparison with Centralized Models

We compared the performance of our FL-based TRN with the existing centralized models for online action detection. Our model achieved results on par with or superior to these models in terms of the mAP scores as shown in Table 1. Moreover, our approach offered distinct advantages in terms of data privacy preservation, scalability, and adaptability.

Table 1. Results of applied methods using mAP (%).

		Method			
		Single	**FedMean**	**FedMedian**	**FedProx**
Individual actions	intersection passing	78.0%	80.5%	**80.7%**	**79.0%**
	left turn	**77.7%**	76.5%	76.6%	77.1%
	right turn	73.8%	77.9%	74.2%	**78.1%**
	left lane change	**51.1%**	46.9%	43.4%	44.7%
	right lane change	37.6%	38.9%	**39.2%**	39.1%
	left lane branch	**46.0%**	44.2%	45.8%	45.9%
	right lane branch	5.7%	5.5%	**6.0%**	**6.0%**
	crosswalk passing	13.3%	11.5%	**14.2%**	13.7%
	railroad passing	**2.8%**	0.9%	1.2%	1.7%
	merge	**7.2%**	3.2%	7.1%	**7.2%**
	u-turn	**51.9%**	32.0%	51.1%	51.5%

Federated learning (FL) using median aggregation and mean training time is approximately 5 h on average per client, with a client size of 4. In contrast, training the centralized model takes 20 h.

The federated approach uses a smaller batch size of 8, which requires a maximum of 2 GB of GPU memory per client. The centralized model, however, uses a larger batch size of 32, requiring up to 8 GB of GPU memory.

These results demonstrate the effectiveness of our FL-based approach in addressing the challenges of online action detection while ensuring privacy and real-time responsiveness.

6. Discussion and Conclusions

In this part, we analyze the implications and insights obtained from our experimental results and evaluate the performance of our federated learning (FL)-based temporal recurrent network (TRN) for online action detection.

6.1. Interpretation of Results

The FL-based TRN presented in our experiments consistently achieves competitive mAP scores, demonstrating its accuracy in real-world action detection and localization. Notably, it exhibits remarkable scalability across edge devices, ensuring its adaptability to dynamic environments while preserving data privacy. The model's efficient real-time processing, generating action predictions within milliseconds, underscores its suitability for time-sensitive applications.

6.2. Advantages and Limitations

Our FL-based TRN offers several advantages, including privacy preservation, scalability, adaptability, and real-time responsiveness. However, we acknowledge certain limitations:

- Communication Overhead: While FL minimizes the data transfer requirements, communication overhead is still involved in aggregating model updates from edge devices. This overhead can affect the responsiveness of the system in low-bandwidth environments.
- Model Complexity: Including FL and a temporal decoder increases the model's complexity, potentially requiring more computational resources during training and inference. Optimization strategies may be necessary for resource-constrained edge devices.
- Edge Device Heterogeneity: Edge devices in a real-world deployment may exhibit varying computational capabilities and data quality. Addressing the heterogeneity of edge devices is an ongoing challenge.

6.3. Practical Implications

Our FL-based TRN has significant practical implications for various domains, including surveillance, autonomous vehicles, and smart cities. The approach's privacy-preserving nature makes it well-suited for applications where data confidentiality is paramount. Its scalability ensures that the model can adapt to evolving deployment scenarios, while its adaptability caters to the dynamic nature of real-world actions.

6.4. Comparison with Existing Models

Compared with the existing centralized models, our FL-based TRN offers a unique blend of competitive accuracy and privacy preservation. This advantage positions it as a valuable solution for organizations and industries that require real-time action detection while safeguarding sensitive video data.

7. Future Directions

As part of our ongoing research, we are exploring several future directions, including the following:

- Optimization Strategies: We are investigating optimization techniques to reduce the communication overhead in FL, enabling more efficient model updates.
- Resource-Aware Models: We are developing resource-aware TRN models tailored for edge devices with varying computational capabilities, ensuring scalability and adaptability in heterogeneous environments.
- Multimodal Integration: We are exploring integrating multiple data modalities (e.g., audio, depth) with TRN to provide a richer context for action detection.

Our FL-based TRN bridges the gap between online action detection and privacy-preserving federated learning, offering a versatile, real-time, and privacy-conscious approach to address the demands of today's dynamic data landscape. It contributes to the advancement in online action detection and holds promise for various applications in data-sensitive and distributed computing scenarios.

Author Contributions: Conceptualization, A.B.; Methodology, A.G.; Software, A.G.; Validation, A.G.; Writing—original draft, A.G. and A.B.; Writing—review & editing, A.B.; Supervision, A.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The HDD dataset is available for non-commercial usage and can be requested from Hoda Research Institute for academic research purposes on https://usa.honda-ri.com/hdd, accessed on 5 June 2024. The TVseries dataset is available on https://homes.esat.kuleuven.be/psi-archive/rdegeest/TVSeries.html, accessed on 5 June 2024.

Conflicts of Interest: Author Alpaslan Gökcen was employed by the company Turkcell İletişim Hizmetleri. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Wang, X.; Zhang, S.; Qing, Z.; Shao, Y.; Zuo, Z.; Gao, C.; Sang, N. OadTR: Online Action Detection with Transformers. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021. [CrossRef]
2. Chen, J.; Mittal, G.; Yu, Y.; Kong, Y.; Chen, M. GateHUB: Gated History Unit with Background Suppression for Online Action Detection. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022. [CrossRef]
3. Xu, M.; Gao, M.; Chen, Y.T.; Davis, L.; Crandall, D. Temporal Recurrent Networks for Online Action Detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019. [CrossRef]
4. Yang, L.; Han, J.; Zhang, D. Colar: Effective and Efficient Online Action Detection by Consulting Exemplars. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022.
5. Kim, Y.H.; Nam, S.; Kim, S.J. Temporally smooth online action detection using cycle-consistent future anticipation. *Pattern Recognit.* **2021**, *116*, 107954. [CrossRef]
6. Xu, M.; Xiong, Y.; Chen, H.; Li, X.; Xia, W.; Tu, Z.; Soatto, S. Long Short-Term Transformer for Online Action Detection. In Proceedings of the Conference on Neural Information Processing Systems (NeurIPS), Virtual, 6–14 December 2021.
7. Geest, R.D.; Gavves, E.; Ghodrati, A.; Li, Z.; Snoek, C.; Tuytelaars, T. Online Action Detection. *arXiv* **2016**, arXiv:cs.CV/1604.06506.
8. De Geest, R.; Tuytelaars, T. Modeling Temporal Structure with LSTM for Online Action Detection. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018. [CrossRef]
9. Gao, J.; Yang, Z.; Nevatia, R. RED: Reinforced Encoder-Decoder Networks for Action Anticipation. *arXiv* **2017**, arXiv:cs.CV/1707.04818.
10. Shou, Z.; Pan, J.; Chan, J.; Miyazawa, K.; Mansour, H.; Vetro, A.; Giro i Nieto, X.; Chang, S.F. Online Detection of Action Start in Untrimmed, Streaming Videos. *arXiv* **2018**, arXiv:cs.CV/1802.06822.
11. Quach, S.; Thaichon, P.; Martin, K.; Weaven, S.; Palmatier, R. Digital technologies: Tensions in privacy and data. *J. Acad. Mark. Sci.* **2022**, *50*, 1299–1323. [CrossRef] [PubMed]
12. Bonawitz, K.A.; Eichner, H.; Grieskamp, W.; Huba, D.; Ingerman, A.; Ivanov, V.; Kiddon, C.M.; Konečný, J.; Mazzocchi, S.; McMahan, B.; et al. Towards Federated Learning at Scale: System Design. *Proc. Mach. Learn. Syst.* **2019**, *1*, 374–388.
13. Hard, A.; Kiddon, C.M.; Ramage, D.; Beaufays, F.; Eichner, H.; Rao, K.; Mathews, R.; Augenstein, S. Federated Learning for Mobile Keyboard Prediction. *arXiv* **2018**, arXiv:1811.03604.
14. Xia, Q.; Ye, W.; Tao, Z.; Wu, J.; Li, Q. A survey of federated learning for edge computing: Research problems and solutions. *High-Confid. Comput.* **2021**, *1*, 100008. [CrossRef]
15. Ramanishka, V.; Chen, Y.T.; Misu, T.; Saenko, K. Toward Driving Scene Understanding: A Dataset for Learning Driver Behavior and Causal Reasoning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
16. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53. [CrossRef]

17. Sherstinsky, A. Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Phys. D Nonlinear Phenom.* **2020**, *404*, 132306. [CrossRef]
18. Lea, C.; Vidal, R.; Reiter, A.; Hager, G.D. Temporal Convolutional Networks: A Unified Approach to Action Segmentation. *arXiv* **2016**, arXiv:cs.CV/1608.08242.
19. Ng, J.Y.H.; Hausknecht, M.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; Toderici, G. Beyond Short Snippets: Deep Networks for Video Classification. *arXiv* **2015**, arXiv:cs.CV/1503.08909.
20. Sultani, W.; Chen, C.; Shah, M. Real-world Anomaly Detection in Surveillance Videos. *arXiv* **2019**, arXiv:cs.CV/1801.04264.
21. Kitani, K.M.; Okabe, T.; Sato, Y.; Sugimoto, A. Fast Unsupervised Ego-Action Learning for First-Person Sports Videos. In Proceedings of the CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011; pp. 3241–3248. [CrossRef]
22. Li, Y.; Ye, Z.; Rehg, J.M. Delving into egocentric actions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015. [CrossRef]
23. Ma, M.; Fan, H.; Kitani, K.M. Going Deeper into First-Person Activity Recognition. *arXiv* **2016**, arXiv:cs.CV/1605.03688.
24. Laptev, I.; Marszalek, M.; Schmid, C.; Rozenfeld, B. Learning realistic human actions from movies. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8. [CrossRef]
25. Wang, H.; Kläser, A.; Schmid, C.; Liu, C.L. Dense Trajectories and Motion Boundary Descriptors for Action Recognition. *Int. J. Comput. Vis.* **2013**, *103*, 60–79. [CrossRef]
26. Wang, H.; Klaser, A.; Schmid, C.; Liu, C.L. Action recognition by dense trajectories. In Proceedings of the CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011. [CrossRef]
27. Simonyan, K.; Zisserman, A. Two-Stream Convolutional Networks for Action Recognition in Videos. In Proceedings of the Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, Montreal, QC, Canada, 8–13 December 2014.
28. Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; Van Gool, L. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In Proceedings of the Computer Vision—ECCV 2016, Amsterdam, The Netherlands, 11–14 October 2016; pp. 20–36. [CrossRef]
29. Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning Spatiotemporal Features with 3D Convolutional Networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4489–4497. [CrossRef]
30. Carreira, J.; Zisserman, A. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017. [CrossRef]
31. Xu, M.; Sharghi, A.; Chen, X.; Crandall, D.J. Fully-Coupled Two-Stream Spatiotemporal Networks for Extremely Low Resolution Action Recognition. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 12–15 March 2018. [CrossRef]
32. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
33. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
34. Donahue, J.; Hendricks, L.A.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Darrell, T.; Saenko, K. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015. [CrossRef]
35. Iqbal, S.; Qureshi, A.N.; Alhussein, M.; Aurangzeb, K.; Javeed, K.; Ali Naqvi, R. Privacy-preserving collaborative AI for distributed deep learning with cross-sectional data. In *Multimedia Tools and Applications*; Springer: Berlin/Heidelberg, Germany, 2023. [CrossRef]
36. Jiang, M.; Jung, T.; Karl, R.; Zhao, T. Federated Dynamic GNN with Secure Aggregation. *arXiv* **2020**, arXiv:cs.CR/2009.07351.
37. Xiao, Z.; Xu, X.; Xing, H.; Song, F.; Wang, X.; Zhao, B. A federated learning system with enhanced feature extraction for human activity recognition. *Knowl.-Based Syst.* **2021**, *229*, 107338. [CrossRef]
38. Doshi, K.; Yilmaz, Y. Federated Learning-based Driver Activity Recognition for Edge Devices. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 3337–3345. [CrossRef]
39. Nabil, M.; Sherif, A.; Mahmoud, M.; Alsmary, W.; Alsabaan, M. Accurate and Privacy-Preserving Person Localization Using Federated-Learning and the Camera Surveillance Systems of Public Places. *IEEE Access* **2022**, *10*, 109894–109907. [CrossRef]
40. Tu, N.A.; Abu, A.; Aikyn, N.; Makhanov, N.; Lee, M.H.; Le-Huy, K.; Wong, K.S. FedFSLAR: A Federated Learning Framework for Few-Shot Action Recognition. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, Waikoloa, HI, USA, 3–8 January 2024; pp. 270–279.
41. Guo, J.; Liu, H.; Sun, S.; Guo, T.; Zhang, M.; Si, C. FSAR: Federated Skeleton-based Action Recognition with Adaptive Topology Structure and Knowledge Distillation. In Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–3 October 2023; pp. 10366–10376. [CrossRef]
42. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated Optimization in Heterogeneous Networks. *arXiv* **2020**, arXiv:cs.LG/1812.06127.
43. Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated Learning with Non-IID Data. *arXiv* **2018**, arXiv:1806.00582.

44. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv* **2016**, arXiv:cs.CV/1602.07261.
45. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255. [CrossRef]