Chapter 5a

Action Recognition in the Wild: Fall Action

The growing global population has led to a rise in the number of falling incidents among elderly individuals with neurological decline. Neurological impairment is a major contributor to the risk of falls, although most of the evidence for this comes from studies into the causes of falls in the elderly [181]. Data derived from multidisciplinary fall consultation surveys indicate that potentially fall-inducing neurological disorders were present in two out of three patients [182]. This highlights the urgent societal and healthcare concerns that pose barriers to an active aging society. Falls during actions are a common cause of severe injury among elderly adults and those with neurological disorders [183]. Analysis of various neurological factors related to falls in adults suggests that monitoring past falling experiences could significantly aid in predicting future falls during daily activities and sports events [184]. Hence, devising plans to monitor and diagnose falls is crucial to prevent the risk of injury, improve quality of life, and reduce the burden of healthcare costs. However, continuous monitoring using medical equipment for patients in the early stages of neurological decline is costly. It is preferable to implement a low-cost and scalable solution for the regular monitoring of

these patients. Figure 5a.1 shows a certain action performed in daily life, while rows 2-4 show an uncertain action (a certain action followed by a fall). Patients in advanced stages of neurological disorders require medical attention with trained doctors, nurses, and medical equipment.

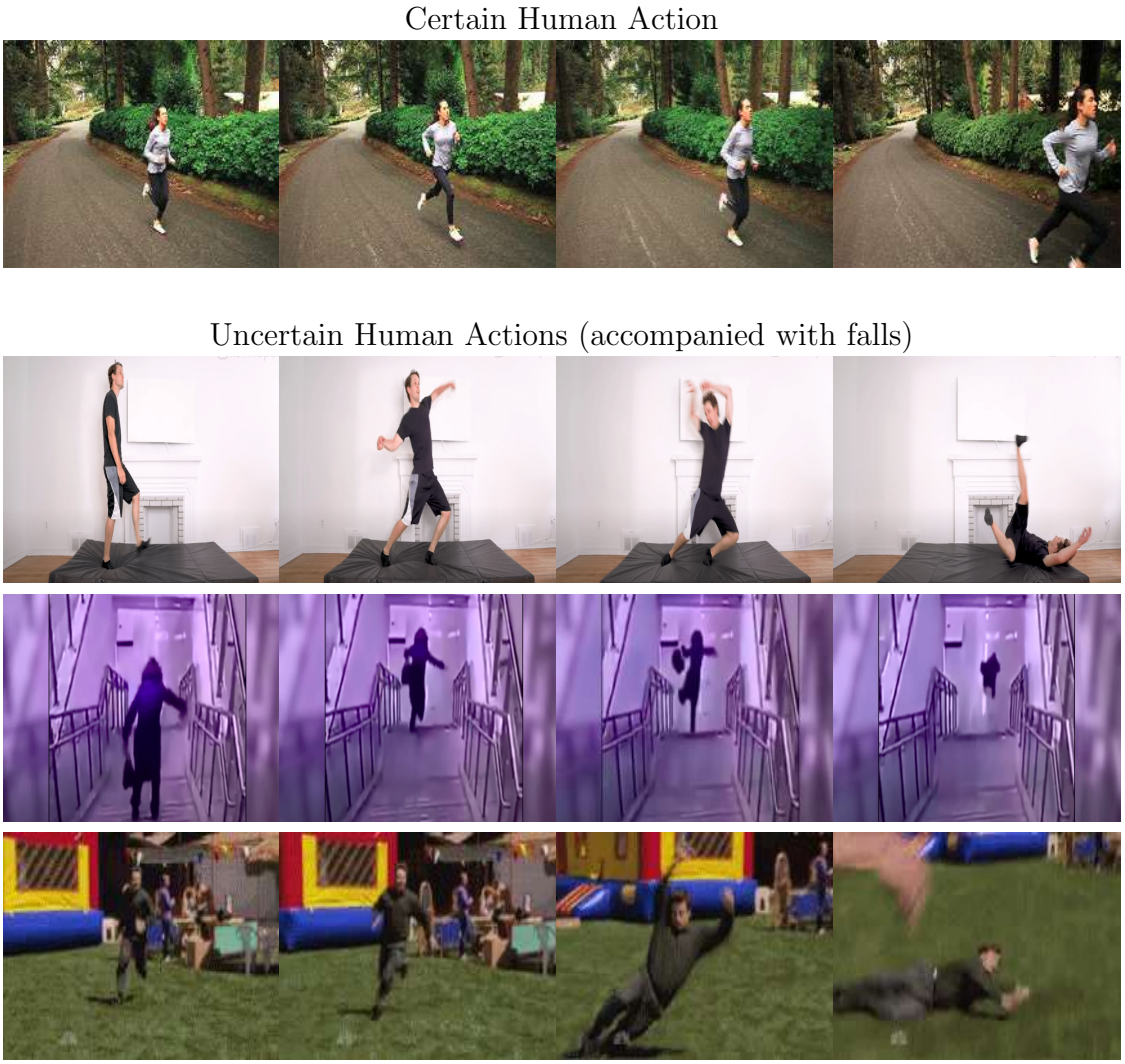


Figure 5a.1: **Certain and Uncertain Human Actions.** A video may consist of actions and falls. The frames are extracted from FallAction and OOPS [185] datasets. First row shows certain action and the last three rows depict certain actions accompanied with fall.

The fall detection system utilizes wearable devices equipped with accelerometers, gyroscopes, switches, or specialized medical equipment to capture rapid movements

during a fall. These sensors trigger alerts upon detection of an uncertain action, however, they require frequent charging and widespread deployment to accurately capture falling events [186]. A camera-based solution, on the other hand, provides an alternative by eliminating charging issues and avoiding any potential negative health effects from the sensors. In computer vision, the aim is to train a model that can perceive visual cues similar to humans. The field of action recognition in computer vision has advanced the understanding of human activities through monitoring and analysis [162, 187–189]. However, detecting uncertain actions remains a challenge due to the risks it poses to a person's wellbeing and safety. Video-based action recognition uses traditional convolutional neural networks (CNN) and recurrent neural networks to extract features from visual data and determine the pose information to differentiate human actions from dominant external factors [190, 191]. However, these models can overlook real-world scenarios that suppress less dominant human activities.

Another challenge in video-based action recognition is acquiring a well-balanced dataset with diverse action classes to train the model without any biases. However, the class distribution of computer vision data is often imbalanced [192], with it being easier to find instances of actions such as "wandering" compared to uncertain actions like falls. This results in a classification problem where the classes are not represented equally, leading to poor performance of conventional algorithms. To overcome this issue, modifications are necessary to prevent the model from simply predicting the majority class in all cases.

The recent advancement of deep neural networks has led to significant progress in resolving the issue of imbalanced classification. Various models [192–195] have been proposed to address this problem, which can be broadly classified into three categories: data resampling-based [193], cost-sensitive learning-based [194], and metric and transfer learning-based [195]. These techniques are implemented as a classifier and integrated with a CNN. However, these methods have some inherent limitations such as the oc-

currence of undesirable noise in oversampling and the loss of relevant information in undersampling. Additionally, metric learning has the potential to result in a biased representation function that is heavily influenced by the majority classes. In short, this analysis addresses the issue of class imbalance, where the number of instances of uncertain actions is much lower compared to instances of certain actions.

Additionally, the our work addresses another challenge, which is the distinct temporal division of uncertain action videos into three parts: certain action, abrupt fall, and post-fall action. In most videos, the abrupt fall and post-fall actions have a shorter temporal duration than the certain action. This leads to weaker inter-class separability between uncertain action videos and certain action videos, exacerbating the problem of unbiased classification. In this work, we posit that instances of uncertain actions are crucial missing elements in action recognition systems.

To address the challenge of uncertain action recognition, we present a novel architecture that leverages semantic supervision through per-class weighting of uncertain actions. This is complemented by a weighted focal loss that emphasizes the learning of feature representations for uncertain actions. The unique advantage of our loss function is its ability to capture better cues of uncertain actions, which is not well captured by traditional loss functions. Our proposed architecture is evaluated on a purely uncertain action dataset (OOPS) [185] as well as on datasets with up to 10% uncertain action instances, such as HMDB51 [2] and Kinetics-600 [196].

5a.1 Our Contribution

The main findings of this work are outlined as follows:

- Recognizing uncertain actions, such as those accompanied by falls, is important for monitoring individuals with brain aging diseases or neurological disorders in order to prevent serious injuries and improve quality of life. This is a relatively under-explored area and presents a significant challenge.

- We present a deep learning model, named as "FallNet", that can classify both definite and ambiguous actions in videos. To include temporal context, we use temporal deformable convolution and four cascaded dilated convolutions, which allows for semantic supervision through per-class weighting of ambiguous actions. This results in the learning of deep, distinguishable features for recognizing ambiguous actions.
- We adopt the concept of learning feature representations of ambiguous actions in videos through a weighted focal loss function. This approach addresses the issue of class imbalance and optimizes inter-class variance while reducing intra-class separability, ensuring fair recognition of both definite and ambiguous actions in videos.
- We conduct extensive experiments to demonstrate the effectiveness of the network. To the best of our knowledge, no dataset in the literature solely annotates falling actions of humans, except the OOPS dataset (which focuses on unintentional actions). Thus, we construct the "FallAction" dataset from online "fail" videos.

5a.2 Organization of the Chapter

The rest of the chapter is summarized as follows: In the next section, we review the existing literature. Section 5a.4 presents our proposed method for recognizing uncertain and certain actions in videos. The ablation study and qualitative results are described in Section 5a.5. The chapter then concludes in Section 5a.6. We finally mentioned the publications related to this chapter.

5a.3 Literature Survey

Action recognition uses spatial information to recognize humans and surrounding objects with their interaction and motion trajectories for representing video and learning

temporal motions. Leveraging uncertain action is used for predicting goals, predicting video speed, video context, and event order [185, 197]. The authors in [185] have investigated mid-level perceptual clues to recognize unintentional action in the video. The model fails to capture the most challenging cases of unexpected interventions, a person in the video has limited visibility, or unintentional action which is caused by environmental factors.

•**Motivation of this Chapter:** The existing deep learning methods in the aforementioned studies (*e.g.*, two-stream CNN, 3D CNN, I3D, *etc.,*) are trained on the data based on daily activities. Furthermore, these methods have challenges recognizing the fall from similar regular actions, such as lying down. Therefore, aforementioned literature are unable to handle falling actions or uncertain actions The aforesaid analysis motivated us to design an architecture that can incorporated fine-grained features to handle uncertain actions. This will help in reducing the confusion of intentional action with the start of failure in a video. Also, the architecture should learn feature representations of uncertain actions without self-supervision.

5a.4 Proposed Approach

In this work, we design a deep neural network for falling action recognition, named as *FallNet*, to facilitate certain and uncertain action recognition. FallNet is divided into three main components: feature extraction module, uncertain-action supervision module, and uncertain-certain action classifier, which are shown in Figure 5a.2. Motivated by the impressive performance of spatio-temporal 3D CNNs and their variants [12,140,198], we have used I3D [140] as our feature extraction module that provides the discriminative features of different video action classes. The uncertain-action supervision module is leveraged to highlight features of uncertain actions using weighted cross-entropy loss. Finally, we extract coarse-level details of certain and uncertain actions present in video using weighted focal loss. The training set of the dataset consists

of a set of pairs $\mathcal{D} = \{x_i, \mathbf{y}_i^C\}_{i=1}^N$, where x_i an instance of a video segment, $\mathbf{y}_i^C$ is corresponding either a certain action category or an uncertain action category, where N represents the number of videos. C represents the number of classes that are further grouped into A certain or U uncertain action sub-categories. Formally, the dataset has partitioned the video indices based on certain and uncertain action sub-categories, represented as: $\{\mathbf{V_a}\}_{\mathbf{a}\in A}$ and $\{\mathbf{V_m}\}_{\mathbf{m}\in U}$. $\mathbf{V}_*$ is the videos and $* \in \{\mathbf{a}, \mathbf{m}\}$ where $\{\mathbf{a}, \mathbf{m}\}$ are the indices denoting the videos of certain classes A and uncertain classes U, respectively.

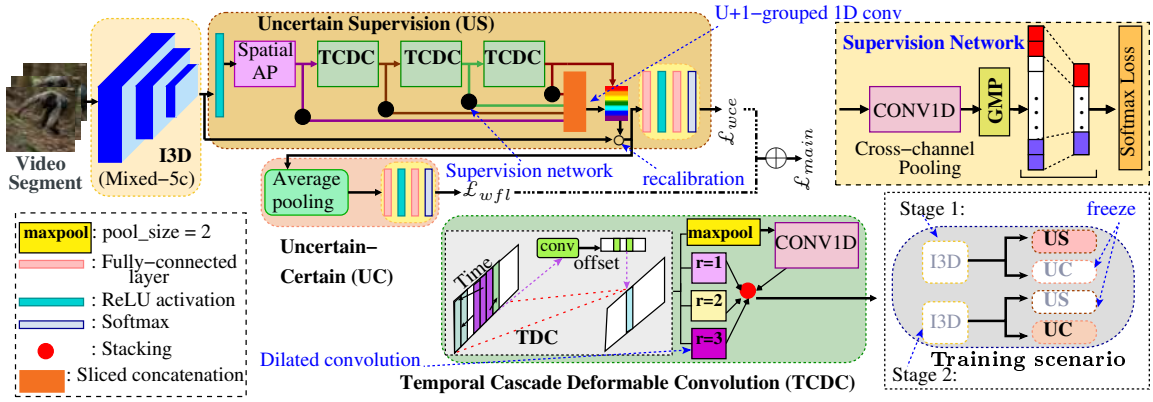


Figure 5a.2: Overview of FallNet. Given an input video segment, FallNet learns the features of certain and uncertain actions present in videos. Spatial AP represents spatial average pooling, GMP is global maxpool, and TDC is temporal deformable convolution layer.

5a.4.1 Feature Extraction Module

This section describes a general formulation for a video feature extractor based on which the proposed classification modules will be presented. Instead of using frame-level feature extractors, we prefer to use 3D networks, which give spatio-temporal features directly from videos. Therefore, we have utilized I3D [140] network (pretrained on Kinetics [133] and OOPS [185] datasets) as our feature extraction function because the features obtained from I3D have larger receptive field and contain more contextual information. A video $\mathbf{V_k}$ is partitioned into successive non-overlapping segments $\mathbf{S}$

and each segment contains an equal number of RGB frames. Formally, the videos are represented as $\mathbf{V_k} = \{\mathbf{S}_t\}_{t=1}^{\mathcal{T}}$, where $\mathcal{T}$ is fixed number of RGB frames in each segment and $\mathbf{k}$ is k-th number of video. The size of input video segments is $\mathcal{T} \times \mathcal{H} \times \mathcal{W} \times \mathcal{C}$, where $\{\mathcal{H}, \mathcal{W}, \mathcal{C}\}$ are height, width, and number of RGB channels ($\mathcal{C} = 3$), respectively. I3D independently processes input video segments to generate encoded representation $\mathbf{O}$ of size $\frac{\mathcal{T}}{8} \times \frac{\mathcal{H}}{32} \times \frac{\mathcal{W}}{32} \times \zeta$, where ζ is the output channels and the value of ζ is 1024. The obtained spatio-temporal feature map is fed into uncertain-action supervision module to predict the confidence score of uncertain action classes.

5a.4.2 Uncertain-Action Supervision (US) Module

This section aims to supervise the discriminative features of different uncertain-action classes present in a video through temporal cascaded deformable convolution (TCDC) layers. TCDC is utilize to expand temporal sampling with temporal dynamics of action learn by the offsets. It is the combination of a temporal deformable convolution layer followed by one maxpool and three multi-scale dilated convolution layers, which are arranged in a parallel fashion. Moreover, the discriminative features are provided as the supervision to uncertain action classifier for unbiased classification.

As we know that the fixed geometric structure of convolution filters largely limits the learning capacity for video action recognition [199], we have therefore introduced a temporal deformable convolution layer to capture the correlation of action features in the temporal domain effectively. In addition, we have appended multi-scale dilated convolutions [200] and a maxpool layer to capture the distinct temporal contextual information. The input spatio-temporal feature map $\mathbf{O}$ is first fed to the ReLU activation function followed by batch normalization to incorporate non-linearity and normalize the values of the feature map. We then squeeze the spatial dimension of the input feature map to adequately sort out relevant temporal cues associated with the specific uncertain action. We compress the spatial dimension using average pooling (AP) to obtain

temporal feature descriptors $\mathbf{F} \in \mathbb{R}^{1\times1\times\mathcal{T}'\times\zeta}$, which is obtained as follows:

$$\widehat{\mathbf{O}} = \mathbf{BN}\ (\mathbf{ReLU}\ (\mathsf{W}_o\ \mathbf{O}), \tag{5a.1}$$

$$\mathbf{F} = \frac{1}{H \times W} \sum_{y=0}^{W-1} \sum_{x=0}^{H-1} \mathbf{ReLU}(\widehat{\mathbf{O}})(y, x), \tag{5a.2}$$

where $\mathbf{BN}(\cdot)$ is the batch normalization, $\mathbf{ReLU}(\cdot)$ denotes ReLU activation function and W_o is learnable parameter. The dimension of $H = \frac{\mathcal{H}}{32}$, $W = \frac{\mathcal{W}}{32}$, and $\mathcal{T}' = \frac{\mathcal{T}}{8}$. $\mathbf{F}$ represent collection of global descriptors expressive over entire video.

Next, temporal deformable convolution is performed in two stages: (i) the estimation of the temporal offset by processing the input pooled features using a 1D convolution; and (ii) the temporal offsets are exploited through extra temporal convolution and aggregated with the output feature obtain in stage 1. Formally, the temporal deformable convolution performs sampling over time constraint $(t_{\mathtt{k}} + \delta t_{\mathtt{k}})$, which is formulated by:

$$\mathfrak{Z}(t_0) = \sum_{\mathtt{k}\in K} \mathsf{W}(t_{\mathtt{k}})\mathbf{F}(t_0 + t_{\mathtt{k}} + \delta t_{\mathtt{k}}), \tag{5a.3}$$

where $\delta = \{\delta t_{\mathtt{k}} | \mathtt{k} = 1, 2, \cdots, |K|\}$ is a set of offsets. W is convolution weights which are learnable parameters and K represents the size of temporal receptive field. Here, $t_{\mathtt{k}} + \delta t_{\mathtt{k}}$ is a real number since $\delta t_{\mathtt{k}}$ is a real number and the nearest integer temporal location $\mathtt{k}$ in the temporal descriptor is calculated through bilinear interpolation in time dimension. $\delta t_{\mathtt{k}}$ is obtained through a temporal convolutional to the input temporal descriptor.

Then, we incorporate maxpool and dilation in vanilla convolution layer to discover temporal correlation within different video segments. We have added one maxpool with kernel 2 followed by one point wise temporal convolution to reduce channels to $\zeta/4$ and stride 1. The output pooled feature descriptor $\mathcal{M} \in \mathbb{R}^{\mathcal{T}'\times\zeta'}$ is obtained from maxpool operation ($\boldsymbol{max}(\cdot)$) with stride (s) as 1 and kernel (k) of size 2 followed by point wise

temporal convolution, which is given by:

$$\mathcal{M} = Conv(\underset{x\in\mathcal{T}',y\in\zeta'}{\boldsymbol{max}}(\mathfrak{Z}_{x,y})), \tag{5a.4}$$

where $\zeta' = \frac{\zeta}{4}$, $\mathcal{M}$ is pooled feature descriptor, and $\boldsymbol{max}(\cdot)$ is maxpool operation. $Conv(\cdot)$ is point wise temporal convolution function. In parallel, we fed temporal feature map $\mathfrak{Z}$ to temporal dilated convolution block with three dilation rates $\boldsymbol{r} \in \{1, 2, 3\}$. In a dilated convolution block, dilation rates are different but kernel $\boldsymbol{W}$ and output channels ζ' remain same for all convolutions. The receptive field $\mathcal{R}$ in a dilated convolution with a temporal kernel $\boldsymbol{W}$ and dilation factor $\boldsymbol{r}$ is calculated as $\mathcal{R} = [(\boldsymbol{W}-1)\boldsymbol{r}+1]$. The size of receptive field $\mathcal{R}$ depends on the choice of the size of the kernel $\boldsymbol{W}$ and the value of the dilation factor $\boldsymbol{r}$. Formally, the output feature map obtained by formulating dilated convolution operation [200] with a given input $\mathfrak{Z}$ and dilated kernel $\boldsymbol{W}$ is computed by:

$$\mathfrak{F}(t) = \mathfrak{Z}(t)\textcircled{\boldsymbol{r}}\boldsymbol{W}(t) = \sum_{i\in\mathcal{T}'} \mathfrak{Z}(i)\boldsymbol{W}(x - \boldsymbol{r}i), \tag{5a.5}$$

where $\textcircled{\boldsymbol{r}}$ is dilated convolution operation with $\boldsymbol{r}$ dilation factor. Finally, we channel-wise stacked the output of convolutions with different dilation factors to produce a multi-scale response, as shown in Figure 5a.2. The stacked features $\mathfrak{X} \in \mathbb{R}^{\mathcal{T}'\times\zeta}$ is calculated by:

$$\mathfrak{X} = \lceil \lfloor \mathcal{M}, \mathfrak{F}_1, \mathfrak{F}_2, \mathfrak{F}_3 \rfloor \rceil, \tag{5a.6}$$

where $\lceil \lfloor \cdot, \cdot \rfloor \rceil$ represents the stacking operation. The reason behind using stacking operation instead of overlapping using element-wise sum operation is to compute, aggregate, and pass information over multiple layers.

We have stacked $\mathcal{N}$ TCDC layers sequentially to grouped the temporal contextual cues of uncertain action. Next, we incorporate supervised classification using uncer-

tain classes, which is motivated through [176]. Since, the features of uncertain classes have poor inter-class variability, we perform multiple supervision over several layers to learn the deep discriminative features of uncertain action class. Therefore, we have incorporated supervision network (SN) that consists of a $1D$ temporal convolution with $m(U+1)$ filters followed by a global maxpool layer (GMP), cross -channel pooling and m-way softmax classification. SN is applied on the output of AP layer and each TCDC layer to produce a $U+1$ discrimative uncertain action features $\{\mathcal{F}_1^i, \cdots, \mathcal{F}_{U+1}^i\}$ where $i \in \{0, \cdots, \mathcal{N}\}$. In other words, we have $U+1$ uncertain classes and each class has m discriminative features which leads to $m(U+1)$ filters in $1D$ temporal convolution. After extracting the maximum values from every filter using maxpool, the $m(U+1)$-dimensional feature map is obtained. Next, we have averaged the values across each group of m-dimensions using cross-channel pooling to obtain $(U+1)$-dimensional feature map. Finally, we fed the pooled feature map into an $(U+1)$-way softmax loss to obtain maximum discrimative features of uncertain classes. We then fuse these $U+1$ feature maps through a sliced concatenation layer [201] to produce a $(\mathcal{N}+1)(U+1)$-channel feature map, as follows:

$$\mathcal{D} = \{\mathcal{F}_1^0, \cdots, \mathcal{F}_1^{\mathcal{N}}, \cdots, \mathcal{F}_{U+1}^0, \cdots, \mathcal{F}_{U+1}^{\mathcal{N}}\}. \tag{5a.7}$$

$\mathcal{D}$ is fed into our fused classification layer which performs $U+1$-grouped $1D$ convolution to produce a $U+1$-channel feature map $\mathcal{P}$. The uncertain action feature descriptor are supervised by the ground truth of the uncertain action ($\boldsymbol{y}^u$). Finally, we recalibrated the obtained fused uncertain action feature descriptor $\mathcal{P}$ with the spatio-temporal feature map $\mathbf{O}$ to reflect the uncertain features on spatio-temporal map, is given by:

$$\mathbf{Q} = [\mathbf{o}_1 + \mathbf{o}_1 \cdot \mathbf{p}_1, \cdots, \mathbf{o}_{\mathcal{T}'\times\zeta} + \mathbf{o}_{\mathcal{T}'\times\zeta} \cdot \mathbf{p}_{\mathcal{T}'\times\zeta}], \tag{5a.8}$$

where $\mathbf{Q}$ is the recalibrated feature map. Based on our experimentation, we consider

$\mathcal{N} = 3$. Thus, five loss functions are computed on $\{\mathcal{F}^0, \cdots, \mathcal{F}^{\mathcal{N}}, \mathcal{P}\}$ to provide deep supervision to our network. We have computed the loss on $\mathcal{P}$, which is formulated by:

$$\pounds(\boldsymbol{y}^u, \mathcal{P}) = -\log(\sum_{i=1}^{U+1} \boldsymbol{exp}(\mathcal{P}_i)) - \mathcal{P}_{\boldsymbol{y}^u}. \tag{5a.9}$$

Similarly, we have computed the loss $\pounds(\boldsymbol{y}^u, \mathcal{F}^{\mathcal{N}})$ on $\{\mathcal{F}^0, \cdots, \mathcal{F}^{\mathcal{N}}\}$, which is computed as:

$$\pounds(\boldsymbol{y}^u, \mathcal{F}^{\mathcal{N}}) = -\log(\sum_{i=1}^{U+1} \boldsymbol{exp}(\mathcal{F}_i^{\mathcal{N}})) - \mathcal{F}_{\boldsymbol{y}^u}^{\mathcal{N}}. \tag{5a.10}$$

The output recalibrated feature map $\mathbf{Q}$ is fed into a softmax which aims to minimize the standard cross entropy loss function. Here, we have use weighted cross-entropy to provide class-wise weights of uncertain action, formulated by:

$$\pounds_{wce} = \frac{1}{N_u} \sum_{i=1}^{N_u} \sum_{j=1}^{U} \pounds_{ij}, \tag{5a.11}$$

$$\pounds_{ij} = -\hat{\mathsf{W}}_{ij} \cdot \boldsymbol{y}_{ij}^u \cdot \log(\mathbf{Z}_{ij}), \tag{5a.12}$$

$$\mathbf{Z} = \textbf{softmax}\ (\mathsf{W}_z \cdot \mathbf{ReLU}(\mathbf{Q}) + \mathbf{b}_z), \tag{5a.13}$$

where N_u is number of training instances of uncertain action, $\boldsymbol{y}^u$ is the ground-truth of uncertain instance and U is number uncertain action classes. The value of $\hat{\mathsf{W}}$ is class-wise weighted map and calculated by:

$$\hat{\mathsf{W}} = \frac{N - \sum_n \mathbf{Z}_n}{\sum_n \mathbf{Z}_n}. \tag{5a.14}$$

This helps in penalizing the US not to consider irrelevant uncertain action classes and assign the higher weights to relevant ones. $\{\mathsf{W}_z, \mathbf{b}_z\}$ are learnable parameters.

5a.4.3 Uncertain-Certain Action (UC) Classifier

To force the model to learn least dominant motion features while training for video-based action recognition, we introduce uncertain-certain action classifier. It simultaneously learns to estimate certain action and uncertain action classes using weighted focal loss. This helps in reducing the class imbalance problem that leads to misclassifcation problem. In uncertain-certain action classifier, first we have obtained temporal feature map and then fed into fully-connected(FC) layers interleaved with ReLU and softmax activation functions, which is formulated by:

$$\mathbf{P} = \textbf{softmax}\ (\mathsf{W}_p \cdot \textbf{ReLU}(\Gamma_u(\mathbf{Q})) + \mathbf{b}_p), \tag{5a.15}$$

where Γ_u is 3D average pooling function with kernel size $1 \times \frac{\mathcal{H}}{32} \times \frac{\mathcal{W}}{32}$. The size of $\widehat{\mathbf{Q}_u} = \Gamma_u(\mathbf{Q})$ is $\frac{\mathcal{T}}{8} \times 1 \times 1$. $\{\mathsf{W}_p\}$ are learnable weights and $\{\mathbf{b}_p\}$ are learnable bias terms. In our case, dataset is imbalanced *i.e.,* the number of action classes are much larger than uncertain action classes due to which uncertain action classes are ignored. It can be handle through conventional focal loss [202], which reduces the relative loss for certain action classified samples and focus on uncertain action samples, mathematical representation is given as follows:

$$\pounds_{fl} = -\sum_{i=1}^{C} \alpha \cdot (1 - \mathbf{P}_i)^{\gamma} \cdot \log(\mathbf{P}_i), \tag{5a.16}$$

where, γ is the focusing parameter that specifies how much higher-confidence correct predictions contribute to the overall loss and $C = A + U$. In other words, higher the γ value, higher the rate at which certain action instances are down-weighted. α is a hyperparameter and $\alpha = 1$ is the default value, *i.e.,* no weighting. However, focal loss underestimate the importance of samples in the classes of concern. In addition, it is sensitive to wrong labeled samples in the training dataset since the wrong-labeled samples might mistakenly be considered as hard samples. Therefore, we have incorporated

the weighting factor which is inversely proportional to number of class-specific uncertain action instances. The standard focal loss is reformulated based on adding weighted factor $\ddot{\mathsf{W}}$, which is estimated by:

$$\pounds_{wfl} = -\sum_{i=1}^{N_c}\sum_{j=1}^{C} \ddot{\mathsf{W}}_{ij} \cdot \mathbf{y}_{ij}^{c} \cdot (1-\mathbf{P}_{ij})^{\gamma} \cdot \log(\mathbf{P}_{ij}), \tag{5a.17}$$

where $\mathbf{y}^c$ is the ground-truth and the weighted factor is calculated on the basis of class score with tuning factor η and number of uncertain and certain action instances N^c, which is formulated by:

$$\mathfrak{S}_j = \log(\frac{\eta \cdot \sum_{i \in U} N_i^c}{N_j^c}),\ \forall j \in U, \tag{5a.18}$$

where boundary conditions of class score with $\eta \geq 1$ is given by:

$$\ddot{\mathsf{W}} = \begin{cases} \mathfrak{S}, & if \quad \mathfrak{S} > 1. \\ 1, & otherwise. \end{cases} \tag{5a.19}$$

The value of score increases weights of sub-classes of uncertain action and helps in balancing sub-classes. We obtained vital information of uncertain actions present in video. However, video also consists of certain action. Thus, we have included certain action classification in our overall loss, which is explained in next subsection.

5a.4.4 Joint Training Model

In training time, we first freeze I3D network and individually train uncertain-action supervision module and uncertain-certain action classifier, respectively. This helps in maximizing performance of each module. We propose three-stage training procedure. [I]. We fine-tune I3D network with both certain and uncertain action instances to extract accurate motion features. [II]. We freeze I3D and train the uncertain-action supervision module with different losses one-by-one. In other words, we first train the

supervision network and then freeze it. Next we train parameters of $U+1$-grouped 1D convolution using the uncertain action classes as the ground-truth. Finally, we freeze the group convolutions and train uncertain-action supervision module using weighted cross-entropy loss. [III]. We train the uncertain-certain action classifier using weighted focal loss while freezing the other two blocks. To jointly recognizing both certain and uncertain actions, we train our proposed architecture in end-to-end manner using main loss. We formulate our main loss $\pounds_{main}$ to converge our architecture by:

$$\pounds_{main} = \lambda_1 \pounds_{wfl} + \lambda_2(\pounds(\boldsymbol{y}^u, \mathcal{P}) + \pounds(\boldsymbol{y}^u, \mathcal{F}^{\mathcal{N}}) + \pounds_{wce}), \tag{5a.20}$$

where $\{\lambda_2(\pounds(\boldsymbol{y}^u, \mathcal{P}), \pounds(\boldsymbol{y}^u, \mathcal{F}^{\mathcal{N}})\}$ are supervision loss, $\pounds_{wfl}$ is weighted focal-loss, and $\pounds_{wce}$ are weighted cross-entropy loss. λ_1, λ_2 are hyperparameters to balance the loss.

5a.5 Experimental Results

In this section, we show comparative study with state-of-the-art action recognition methods and carry out ablation study to evaluate the effectiveness of our architecture. These experiments are performed on NVIDIA Geforce GTX 1080Ti GPUs with i7-6800K CPU. The models are implemented with Pytorch platform [1]. We have utilized accuracy (Acc.) for OOPS, HMDB51, and FallAction [2] datasets and Top-1 and Top-5 metrics for Kinetics-600 dataset.

5a.5.1 Dataset details

In our experiment, we have performed experiments on real-world and publicly accessible video-based uncertain action recognition benchmark dataset, OOPS [185]. As we have only one dataset that purely-based on uncertain action, therefore we have also include the dataset with overlapping of atmost 10% uncertain action that demonstrate

[1] https://github.com/ Nigam-Niti/ FallNet
[2] https://figshare.com/ articles/ dataset/ FallAction_dataset_zip/ 17157845

fall classes. It includes HMDB51 [2] and Kinetics-600 [196]. We have also collected the videos regarding uncertain action named as "FallAction" datasets and reported the experimental results. **OOPS.** It is a large collection of videos containing intentional and unintentional action. It contains 20k videos of daily human activities with unexpected failed moments, among them 14k videos have the timestamps of the failed moments. **Kinetics-600.** It has 392k training videos and 30k validation videos with 600 classes. We report top-1 and top-5 classification accuracy. It has 2 fall classes, i.e., falling_off_bike and falling_off_chair. **HMDB51**. It consists of 50 action classes and 1 uncertain action class. It includes 6,766 videos data and each class contains a minimum of 101 clips.

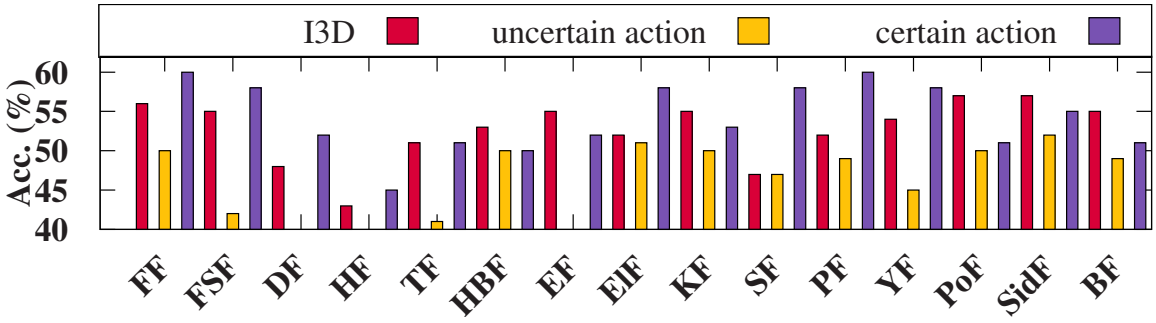


Figure 5a.3: Performance of different modules of our architecture on 15 pair of classes in FallAction dataset. **FF** is Faint_fall, **FSF** is False_start_fall, **DF** is Drunk_fall, **HF** is Handstand_fall, **TF** is Trust_fall, **HBF** is High_Heel_Break_fall, **EF** is Escalator_fall, **ElF** is Elderly_fall, **KF** is Karate_fall, **SF** is Surfing_fall, **PF** is Parkour_fall, **YF** is Yoga_fall, **PoF** is Pool_fall, **SiF** is **SidF** is Side_fall, and **BF** is Butt_fall.

FallAction. Actions performed by a person facing neurological disorder in daily-life is erratic, for instance, 'faint fall'. We build our dataset by collecting videos from online. FallAction provides variation of video classes that are based on certain and uncertain actions. The dataset contain 15 uncertain action categories and each category has an average of 50-100 videos. It is a dataset with 20 uncertain action sub-categories and every class contains an average of 100 videos. The spatial resolution of videos ranges from 360×360 to 640×640. Each trimmed video has an average length of 10 seconds that are taken from the online resources such as YouTube and Movies.

The minimum and maximum resolution of frame is 480 and 1080, respectively. Since dataset is collected from an online source, therefore it includes variations *i.e.*, variations in camera motion, size of objects, background, illumination, and viewpoints. **Study of Recent Methods on FallAction.** We have shown results of different state-of-the-art methods in Table 5a.1. It is clear from results that our architecture outperforms other approaches. Moreover, we also calculate F1 score for imbalanced classification on FallAction dataset. The results shows that our main loss (5a.20) is able to handle imbalance classification problem efficiently due to incorporation of the focal loss. **Study on different classes of FallAction.** We have validated effectiveness of individual modules of our architecture on each action class of FallAction, as shown in Figure 5a.3. It is noted that particular action have different score values for different modules. We observed that US works better in case of 'faint fall' and 'side fall' actions as compared to other modules, as most of actions are focused on temporal context. Similarly, UC works better in 'running' and 'walking' pair of classes. **Confusion matrix of FallAction.** Table 5a.2 shows confusion matrix for some FallAction dataset action classes performed by FallNet. We observe that without uncertain action classifier, our model predict wrong classes.

5a.5.2 Implementation Details

This section describe details related to input, data augmentation, optimization parameters, inference related attributes. **Input size and data augmentation.** We resize all frames of videos to spatial resolution of 224×224 before feeding them into the I3D backbone network. For training, we have downsampled videos at 15 fps on FallAction and 15 fps on OOPS datasets, respectively. Similarly, for HMDB51 the fps is 30. The temporal length of 16 distinct frames for action video is selected for every datasets. In case where the videos are not sufficiently long, we have padded zeros. To fit each batch into GPU memory, we set training batch size to 32. **Backbone.** I3D model that we

Table 5a.1: Performance of current state-of-the-art methods on FallAction dataset. WCE is weighted cross entropy loss, CE is cross-entropy loss, and W-focal loss is weighted focal loss. Acc. denote classification accuracy and F1 is F1-score.

Methods	**Pretrain**	**Loss func.**	**Acc.**	**F1**
C3D [12]	OOPS	CE	60.0	55.7
		WCE	62.2	56.1
		focal loss	62.5	59.0
		W-focal loss	62.9	59.5
		Main loss (5a.20)	63.1	59.9
C3D [12]	OOPS+Kinetics	CE	68.3	57.4
		WCE	68.95	57.8
		focal loss	69.1	59.8
		W-focal loss	70.3	60.0
		Main loss (5a.20)	70.9	60.2
I3D [140]	OOPS	CE	60.0	57.9
		W-CE	62.0	58.2
		focal loss	66.2	60.8
		W-focal loss	66.8	61.1
		Main loss (5a.20)	67.0	61.5
I3D [140]	OOPS+Kinetics	CE	60.3	57.7
		WCE	62.7	58.9
		focal loss	67.5	61.0
		W-focal loss	68.0	61.6
		Main loss (5a.20)	68.1	62.2
FallNet	OOPS	CE	68.4	57.5
		WCE	69.3	58.0
		focal loss	73.0	62.8
		W-focal loss	73.8	63.1
		Main loss (5a.20)	74.3	64.2
FallNet	OOPS+Kinetics	CE	69.3	58.0
		WCE	70.2	58.9
		focal loss	74.1	63.2
		W-focal loss	74.8	64.5
		Main loss (5a.20)	**75.5**	65.0

Table 5a.2: Confusion matrix of FallAction dataset. We have used abbreviations of FallAction classes. **ElF**- Elderly_fall, **SidF**- Side_fall, **YF**- Yoga_fall, **HBF**- High_Heel_Break_fall, **DF**- Drunk_fall, and **FF**- Faint_fall.

ElF	75.3	24.7	0.0	0.0	0.0	0.0
SidF	28.7	71.3	0.0	0.0	0.0	0.0
YF	0.0	0.0	70.8	29.2	0.0	0.0
HBF	0.0	0.0	26.6	73.4	0.0	0.0
DF	0.0	0.0	0.0	0.0	80.9	19.1
FF	0.0	0.0	0.0	0.0	18.7	81.3
	Elf	**SidF**	**YF**	**HBF**	**DF**	**FF**

have used as backbone is first initialized by training on Kinetics and OOPS datasets. We adopt I3D network for feature extraction as the obtained feature have larger receptive fields and contain more contextual information. **Optimization parameters.** We use weights of I3D to initialize parameters of our uncertain-action supervision module and uncertain-certain action classifier, respectively. Adam optimizer to train the our overall architecture. The value of β_1 and β_2 are set to 0.9 and ϵ is 10^{-5}. We have used the polynomial-decay strategy and learning rate is decayed from 0.001 to 0.00001. Experiments show that 100 epochs are enough to converge the model. **Inference.** In inference time, we follow standard procedures which is employed in the literature when comparing to other state-of-the-art. We use 10 regularly-sampled segments with 16 frames in a segment.

5a.5.3 Ablation Study and Qualitative Analysis

In this section, we experiment with our architecture on FallAction, OOPS, HMDB51, and Kinetics-600 datasets. We present an ablation study on the components of proposed architecture. We also show visual results with respect to uncertain and certain actions. **Effect of different components.** We investigate the effect of components present in FallNet to understand the contribution of each in the overall accuracy. The results are reported in Table 5a.3 in terms of accuracy. The top row reports the results of I3D (trained on Kinetics and OOPS), which is our baseline for both OOPS

Table 5a.3: Performance of individual stage of our architecture on FallAction, OOPS, and HMDB51 datasets.

Stages	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
I3D (OOPS)	67.0	64.2	71.0
I3D (Kinetics)	67.4	64.8	71.8
I3D (Both)	68.1	65.2	72.5
I3D + US (OOPS)	68.5	67.8	73.8
I3D + US (Kinetics)	68.9	67.4	74.0
I3D + US (Both)	72.7	69.7	74.8
I3D + UC (OOPS)	71.0	68.5	73.0
I3D + UC (Kinetics)	71.8	69.0	73.2
I3D + UC (Both)	72.1	69.7	75.0
FallNet (OOPS)	74.3	70.6	75.9
FallNet (Kinetics)	74.0	75.4	76.8
FallNet (OOPS + Kinetics)	**75.5**	**77.2**	**77.7**

Table 5a.4: Impact on computational complexity on FallNet for FallAction, OOPS, and HMDB51 datasets.

Models	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**	**params**	**FLOPs**
I3D	68.1	65.2	72.5	12M	27.8G
I3D + Spatial Average Pooling	69.2	66.5	72.9	262K	27.8G
I3D + Spatial Average Pooling + Supervision Network	72.4	67.5	73.0	293K	27.8G
I3D + Spatial Average Pooling + Supervision Network + TCDC	75.0	77.05	76.9	4.0M	28.5G
Overall	**75.5**	**77.2**	**77.7**	16.79M	29.9G

Table 5a.5: Comparison of computational complexity (pretrain with SOTA methods) for FallAction, OOPS, and HMDB51 datasets; * indicates Retrained by ourselves, SF is slow-fast network, and O+K is OOPS+Kinetics.

Models	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**	**pretrain**	**FLOPs**	**params**
I3D*	68.1	65.2	72.5	OOPS	27.8	12.0M
I3D-ResNet50*	70.1	63.0	72.4	ImageNet	335.3	-
I3D-ResNet101*	71.2	63.1	72.6	ImageNet	654.4	-
SlowFast-ResNet101-8×8*	73.1	65.7	73.0	None	125.5	53.7M
X3D-L*	73.5	62.0	71.9	None	**24.8**	**6.1M**
FallNet	**75.5**	**77.2**	**77.7**	O+K	29.9	16.79M

Table 5a.6: Classification accuracy of different base network on FallAction, OOPS, and HMDB51 dataset; O+K is OOPS+Kinetics.

Methods	**Pretrain**	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
I3D	-	60.1	61.4	60.1
3DResNet18	Kinetics	63.0	61.9	61.3
3DCNN	Sports-1M	64.2	62.9	62.7
I3D	OOPS	67.0	64.2	71.0
I3D	Kinetics	67.4	64.8	71.8
FallNet	O+K	**75.5**	**77.2**	**77.7**

Table 5a.7: Comparison of dilation rate in TCDC layers on FallAction, OOPS, and HMDB51 datasets.

TCDC	**Dilation rate**	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
(1)	$\{1\}$	66.1	69.1	70.6
(2)	$\{1, 2\}$	67.3	71.8	71.0
(3)	$\{1, 3\}$	68.9	72.6	72.6
(4)	$\{1, 4\}$	70.8	74.5	72.7
(5)	$\{1, 2, 3\}$	75.7	**77.2**	**77.7**
(6)	$\{1, 2, 4\}$	**75.9**	75.5	72.7

Table 5a.8: Effect of different losses of FallNet on FallAction, OOPS, and HMDB51 datasets.

Loss functions	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
Cross-Entropy loss	69.3	62.9	64.5
Focal loss	74.1	68.9	69.3
W-Cross-Entropy loss	70.2	63.5	66.5
W-Focal loss	74.8	71.3	70.8
Main loss (5a.20)	**75.5**	**77.2**	**77.7**

Table 5a.9: Impact of Max-pooling in TCDC layers on FallAction, OOPS, and Kinetics-600 datasets.

TCDC Variation		**FallAction Acc.**	**OOPS Acc.**	**Kinetics-600 Top-5**
(1)	{M, 1,2,3}	**75.5**	**77.2**	**96.2**
(2)	{1,2,3}	75.0	76.1	95.1
(3)	{1,2,3,4}	73.6	75.4	93.7
(4)	{1,2,3,4,5}	70.3	71.7	90.0

Table 5a.10: Effect of supervision network in our architecture on FallAction, OOPS, and HMDB51 datasets.

Supervision Variations	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
No supervision	68.3	62.3	64.0
Single supervision	74.0	67.9	69.2
Multiple supervision	**75.5**	**77.2**	**77.7**

Table 5a.11: Effectiveness of FallNet in balancing the data.

Balancing techniques	**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
oversampling	72.0	64.2	61.3
undersampling	74.2	66.1	63.0
standard focal loss	75.0	75.2	74.8
Main loss (5a.20)	**75.5**	**77.2**	**77.7**

Table 5a.12: Comparison of different aggregation strategies in TCDC layers on FallAction and OOPS datasets.

Aggregation Strategies		FallAction Acc.	OOPS Acc.
(1)	element-wise product	69.1	70.1
(2)	weighted element-wise product	70.6	73.7
(3)	element-wise sum	72.8	75.6
(4)	weighted element-wise sum	73.6	76.8
(5)	stacking	**75.5**	**77.2**

and FallAction, and HMDB51 datasets. The uncertain-action supervision module (US) is added to the I3D network. Finally, uncertain-certain action classifier (UC) added that achieve the best performance. We observe that each component is important to improve the baseline performance. We have also explore the computational complexity and number of parameters of different components in FallNet in Table 5a.4. We have reported the FLOPs of existing methods in Table 5a.5 to compare the complexity of FallNet with our work. We have retrained the existing methods for fair comparison. We conclude that FLOPs of FallNet is less than other methods except X3D-L model as it doesnot contain extra supervision network. However, the accuracy of our FallNet is better than X3D-L because of temporal deformable convolution that reduces the intra-class variance and provide discriminative features of both certain and uncertain actions. **Different backbone network.** We evaluate I3D network, which is trained from scratch on FallAction and OOPS dataset using class-specific weighted focal loss. We also assess I3D pretrained on Kinetics for action recognition, which is frozen until indicated. In addition, we utilize 3DCNN [12] and 3DResNet18 [203] methods pretrained on Sports-1M and Kinetics to report the classification accuracy. In Table 5a.6, we report the classification accuracy of different methods on FallAction, OOPS, and HMDB51 datasets, respectively.

Impact of TCDC layers. Table 5a.7 demonstrate the comparison results for different dilation rates. Our experiments show that TCDC achieves a better accuracy to

extract the context cues. We conclude that single dilated rate is not sufficient to extract dense and discriminative contextual information of uncertain actions. Table 5a.9 shows the results of using maxpool layer with dilated convolutions on FallAction, OOPS, and Kinetics-600 datasets in terms of accuracy and Top-5. To restrict the number of training parameters, we have limited to three values of dilation rate (maximum). We study the usage to stacking the feature maps instead of element-wise sum operation in TCDC layers as depicted in Table 5a.12. We have further experimented for optimal number of TCDC layers in FallNet architecture to achieve high accuracy in uncertain action recognition. Table 5a.13 elaborate performance results for appending multiple TCDC layers. Initially, we observe that as we increase the number of TCDC layers the performance improves, however computational complexity also increases. When we append more than 3 TCDC layers, accuracy does not improve further, rather with larger number of layers, performance degraded. This is due to overfitting of parameters.

Table 5a.13: Comparison on number in TCDC layers on FallAction, OOPS, and HMDB51 datasets.

TCDC Layers		**FallAction Acc.**	**OOPS Acc.**	**HMDB51 Acc.**
(1)	1	69.1	69.3	70.1
(2)	2	70.2	73.6	75.6
(3)	3	75.5	**77.2**	**77.7**
(4)	4	**75.9**	75.6	75.1
(5)	nil	52.6	52.6	50.9

Table 5a.14: Comparison with state-of-the-art methods on OOPS dataset.

Methods	**Acc.**
C3D [12]	58.0
OOPS!(self-supervised) [185]	61.6
OOPS!(supervised) [185]	64.0
Ours (OOPS)	70.6
Ours (Kinetics)	75.4
Ours (OOPS + Kinetics)	**77.2**

Table 5a.15: Comparison with recent state-of-the-art methods on Kinetics-600 benchmark dataset.

Methods	**Top-1**	**Top-5**
I3D [140]	73.6	-
AttentionNAS [204]	79.8	94.4
LGD-3D R101 [160]	81.5	95.6
TimeSformer-L [205]	82.4	96.0
Ours	82.7	**96.2**

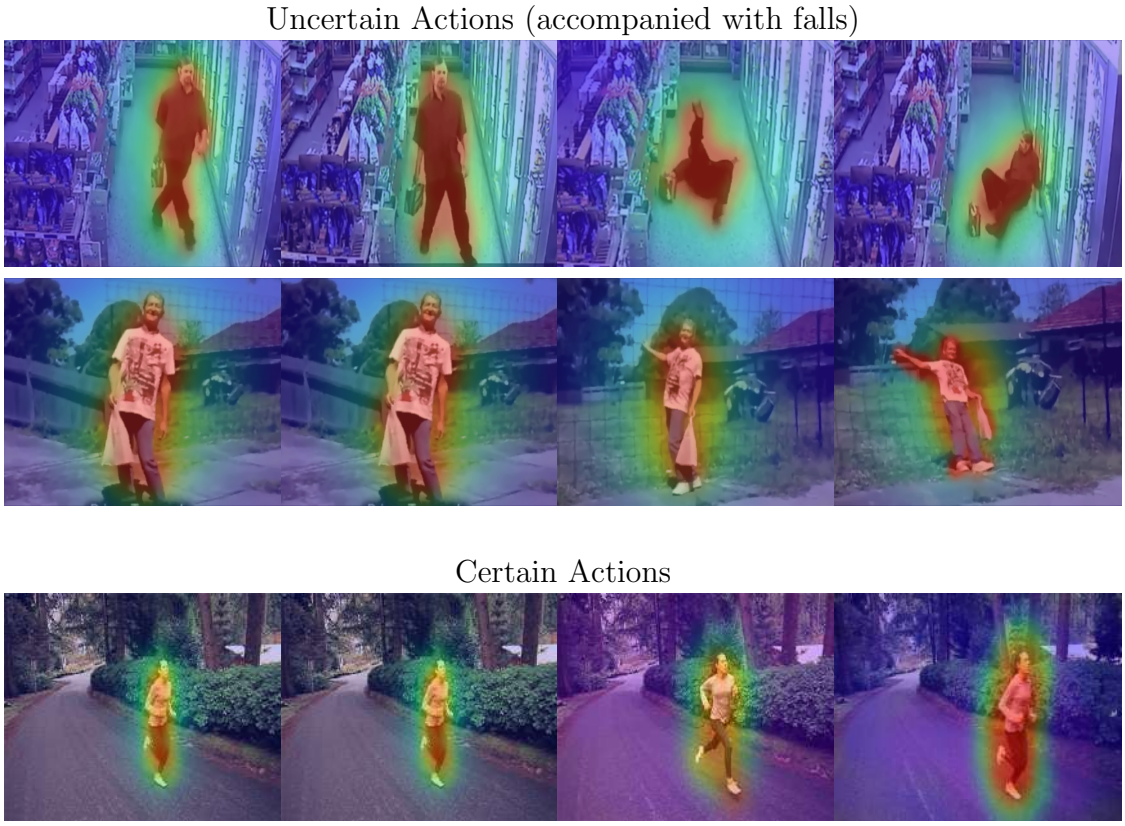


Figure 5a.4: Class activation maps for certain and uncertain action classes.

FallNet effectiveness in balancing data. We show the effect of different techniques for balancing data in terms of performance of our architecture. Table 5a.11 shows the effect of using our focal loss instead of oversampling, undersampling and standard focal loss. We observe that our focal loss is more meaningful and interpretable as it provide mapping between feature maps and uncertain action classes. **Effect of supervision network.** We have shown effect of supervision network in terms of performance of our architecture. Table 5a.10 shows effect of providing supervision that helps in enhancing and learning deep discriminative features of uncertain action. **Effect of different losses.** We have shown effect of different losses in terms of performance of our architecture. Table 5b.5 shows effect of providing different variations of losses and it shows overall loss perform better than individual losses. **Class activation map visualization on FallAction dataset.** We demonstrate efficacy of our uncertain ac-

tion classifier and show results in class activation map (CAM) [206] in Figure 5a.4. We show CAM overlayed over few frames of each input video segment. We present results on FallAction dataset.

5a.5.4 Comparison with State-of-the-Art

We compare FallNet with the state-of-the-art methods in Table 5a.16 and Table 5a.14 on HMDB51 and OOPS datasets, respectively. On OOPS datasets, our proposed model achieved an accuracy of 77.2% and as compare with other related methods we gain an improvement of 13.2%. Moreover, we observed that FallNet is capable of capturing falls due to incorporation of fine-grained information. FallNet provides comparable results with the state-of-the-art literature. It achieves accuracy of 77.7% that beats MVFNet with accuracy of 75.7%. We have also compared different pretrain dataset of our proposed model in Table 5a.16 and it is clear that FallNet outperforms recent approaches on HMDB51 dataset with an improvement of 2.0% for RGB modality. We have also compare with recent methods on Kinetics-600 dataset, as shown in Table 5a.15 and FallNet outperforms previous approaches for Top-5 metric.

Table 5a.16: Comparison with state-the-art methods on HMDB51 dataset.

Methods	**Backbone**	**Pretrain**	**Acc.**
C3D [12]	3D VGG-11	Sports-1M	51.6
STC [145]	ResNet101	Kinetics	66.8
ECO [143]	BNInception+ 3D ResNet-18	Kinetics	72.4
I3D [140]	3D InceptionV1	Kinetics	74.3
TSN [37]	ResNet-50	ImageNet	54.7
TSM [159]	ResNet-50	ImageNet+Kinetics	70.7
STM [207]	ResNet-50	ImageNet+Kinetics	72.2
MVFNet [208]	ResNet-50	ImageNet+Kinetics	75.7
Ours	I3D	OOPS	75.9
Ours	I3D	Kinetics	76.8
Ours	I3D	OOPS+Kinetics	**77.7**

5a.6 Conclusion of the Chapter

The neurocognitive factors have a significant impact on a person's elevated risk of falling. Frequent falls, especially during uncertain actions, are a common cause of injury among elderly adults and individuals with neurological disorders. Monitoring a patient in the early stages of a neurological disorder through continuous medical monitoring can be costly. An "uncertain action" classification model can be a less expensive and scalable alternative for regular monitoring of patients with neurological decline and frequency of relapse.

In this work, we present an efficient deep neural network for semantic supervision of falling actions. Our architecture effectively distinguishes between uncertain and certain actions using weighted loss functions. We have designed a joint training model to train the entire architecture in an end-to-end manner. We have conducted extensive ablation studies to evaluate the performance of our proposed architecture on publicly available benchmark datasets such as HMDB51, Kinetics-600, and OOPS. Our architecture outperforms recent methods on these datasets.

5a.7 Publication related to the Chapter

1. Nitika Nigam, Tanima Dutta and Deepali Verma, "Fall-perceived Action Recognition of Persons with Neurological Disorders using Semantic Supervision", in IEEE Transactions on Cognitive and Developmental Systems (Early Access), doi: 10.1109/ TCDS.2022.3157813.

Chapter 5b

Action Recognition in the Wild: Unusual Action

Uncertainty in behaviour, physical infirmity in action, and memory lapses are periodic among people suffering from neurological illnesses, especially in aged elders. Wearable IoT devices, including smartwatches, smartbands, smart clothes, and jewelry, use sensors such as gyroscopes, accelerometers, and proximity sensors. However, these devices have some limitations including the need for frequent battery replacements, dependence on their environment, and potential inaccuracies in readings.

To address aforesaid limitations, alternative solutions such as implanted devices or improved wearable technology may be developed in the future. An alternative solution for round-the-clock monitoring is the installation of surveillance cameras in the patient's home and portico. This option eliminates the challenges of wearable IoT devices, such as the need for battery replacements, environmental dependencies, and inaccurate readings. It also avoids the issue of forgetting to wear the device, as well as any potential health risks related to sensor radiation. This solution may offer a more cost-effective option in the long term compared to wearable IoT devices. The use of IoT-based automated systems in healthcare poses a threat to the privacy and security

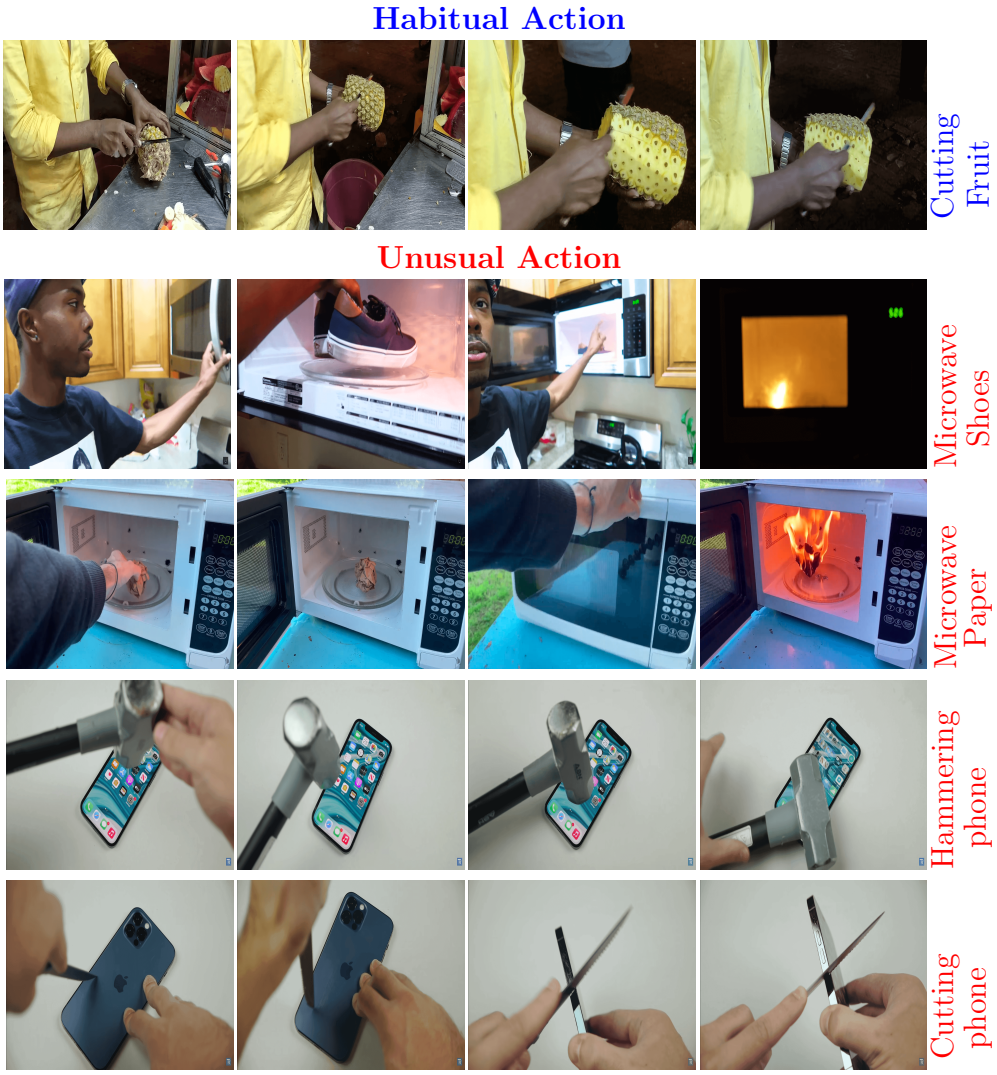


Figure 5b.1: **Habitual vs. Unusual Action.** A video may consist of normal and abnormal actions that depicts the behaviour of a human. The frames are extracted from the RareAct and our '**UnusualAction**' dataset. In the first row, a habitual action is depicted, and unusual actions that could result in an accident are shown in the last four rows.

of patients' sensitive and critical data. This can lead to security breaches. To address these concerns, a video-based smart surveillance system is a preferred solution for continuously monitoring individuals with mild cognitive impairments. This system can identify unusual behaviors and activities and alert families or caregivers if necessary. However, current video-based systems only classify unusual falls and do not recognize complex actions involving human-object interactions. Additionally, these systems do not address privacy concerns surrounding the patients' data. For instance, a patient uses a detrimental combination of two common objects, such as metal or inflammable objects (like shoes or paper) in a microwave oven, that may lead to an accident and cause fire, as shown in Figure 5b.1 (second and third rows).

A video-based smart surveillance system is preferred for continuously monitoring individuals with mild cognitive impairments (MCI) to identify unusual behavior and generate alarms for caregivers or family members if necessary [209]. However, existing video-based systems only classify falling activities and do not recognize complex actions involving human-object interactions. Additionally, they do not address the privacy concerns of sensitive patient data.

The incorporation of federated learning concepts [210–213] over the edge devices may help to train artificial intelligence (AI) models without anyone seeing or touching the patients information and can prevent from the risk of a security breach. These edge devices are smart but low-cost in nature, i.e., smartphones and tablets. Recently, many privacy-protected problems in vision-based applications, such as image classification [214], object detection [215,216], and video analysis [217] are resolved via federated learning. Though the existing works on federated learning have develop models for handling the data distributed drift issue either locally or globally, however a solution bridging the gap is still in research.

Video-based action recognition is a trending area in the field of computer vision. It is helpful in various applications, such as assistance in human-robot interaction [218],

video surveillance [219], human emotion recognition [220], and monitoring of elderly ones [221]. There are various deep neural network (DNN) models [11–13, 37, 54, 78, 140, 222] that help to recognize the type of action performed by a person. To extract the spatial information, existing 2D convolutional neural network (CNN) methods [54] classify actions based on RGB frames. In [11,140], the addition of optical flow is needed for further enhancement of performance. Recent techniques based on 3D CNNs directly extract spatiotemporal features from a video [13,140]. In the meantime, networks learn from action's long-term temporal dynamics are explained in [37, 78, 222, 223].

5b.1 Challenges in this work

The authors identify several challenges in using existing deep neural network models (CNNs) for recognition of unusual human behaviors. These challenges include: (1) limited computing capacity and data present on edge devices, (2) high heterogeneity in data distribution on edge devices, (3) privacy and security concerns with sharing sensitive data, and (4) difficulty in learning representations for rare and unusual actions, which are given by:

1. The existing popular action recognition models are well trained on day-to-day habitual/ usual actions present in benchmark datasets. Direct utilization of popular pre-trained models on habitual actions perform biased classification for unusual actions even after fined-tuned on small dataset, like RareAct, due to the presence of subtle difference in human-object interactions between the videos consisting of habitual and unusual actions. Therefore, the first research challenge is **how to recognize and learn the distinct and finegrained features of unusual action from a given video in an unbiased manner?**
2. Privacy issues are important when there is a chance of leakage of critical data of patients. People with MCI perform unusual actions. It is thus important to protect the data of patients involved in unusual recognition process. The next

challenge is to **how to recognize unusual behaviour of patients suffering from MCI and generate real-time alerts addressing the privacy concerns?**

3. The current methods for privacy-secured learning, such as federated learning, rely on optimizing a shared prediction model between the central server and the diverse edge devices. The heterogeneity of local data distribution across edge devices (clients) is a major obstacle in federated learning. A better feature representation can be extracted by the global model trained on the entire dataset than by the local model trained on a skewed subset. Additionally, the model learns a poorer representation during the local training phase due to the skewed local data distribution. In the local updates, this results in a drift or covariate situation. **Controlling the drift and bridging the representational gap** is another obstacle.
4. We found RareAct [224] video dataset closely related to our problem of unusual action. It depicts rare or unusual actions in the form of noun and verb combinations. However, the number of classes and data per class is very small in nature. Also, RareAct rely on a noun and a verb combination and ignore the combination of unusual pairs of objects. Therefore, the last research question that arises is **how to create a large and balanced videos dataset of unusual actions with human-object and object-object pairs?**

5b.2 Our Contribution

The authors propose a new architecture called "UnusualFedNet" for unusual action recognition. This architecture consists of two components: the UBNet, which is a light-weight model that helps to learn the spatiotemporal feature representations of unusual actions, and the federated contrastive learning (FCL) model, which considers the limited computing capacity and data present on edge devices, and allows for easy

training and inference on these devices with high heterogeneity. The chapter highlights four main contributions of this new architecture:

1. In this work, we develop a low-cost, privacy secured, and vision-based smart surveillance system to tackle a rarely explored problem of recognizing patient's unusual behaviours, who are suffering from mild cognitive declination, over decentralized edge devices holding local data samples, without exchanging them via federated contrastive learning. Our framework is robust, fast, and light-weight in nature.
2. We propose a novel fine-grained human-object relation module that encourages to learn feature representations of relevant information related to unusual combination of human and object that occur rarely in real-world scenario. We also incorporate novel unusual affordance loss to enhance the generalization performance of interaction module through learning the relationship constraint between unusual human-object relation.
3. To design a privacy secured model for patient's data, we develop a light-weight network that can run on decentralized and heterogeneous edge devices that can detect unusual actions. We also added instance contrastive loss to handle the heterogeneity of local data distribution across clients. Under different data scenarios, it can control the drift and bridge the gap between the representation.
4. We demonstrate our experimented results on RareAct [224] and our 'UnusualAction' datasets. RareAct which has almost 20% overlapping classes with our dataset. For fair comparison, we also perform an exhaustive set of experimentation on benchmark habitual action datasets to show the efficacy of the network.

5b.3 Organization of the Chapter

This chapter contains a summary of existing research in the field of video unusual action recognition, followed by a proposed method for recognizing unusual actions in

videos. The effectiveness of this method is then studied through an ablation study and qualitative findings are discussed. The chapter concludes finally in Sections 5b.7 and 5b.8.

5b.4 Literature Survey

The most recent research on HOI-based, unusual, and federated learning-based action recognition is summarized in this section.

• **HOI-based habitual action recognition:** Many methods have been proposed with the recent renaissance of deep networks and the availability of large-scale HOI dataset [225, 226]. Nitika et. al. have proposed a deep network that focused on reducing the ambiguity of actions occur due to context and co-occur objects [162]. To learn more effective HOI representations, some recent leading approaches made use of pose cues [225] or tried to automatically discover informative human parts [226]. In [227], trainable graph neural network (GNN) is exploited to learn the interaction between objects and actions from images and videos. It required additional interactions annotations for action recognition. The authors in [228] exploited videos to extract human-object interaction hotspots with weak supervision modes.

• **Unusual action recognition:** Most of deep networks focus on recognizing and detecting fall actions [209, 229] and abnormal behaviour recognition [3] from videos. However, unusual actions are not taken in consideration till now in videos. Some works have been done for detecting unusual relationship in images [230, 231]. For example, an unusual action like 'a baby is sitting on a dog' have the triplets person, ride, and dog, indicate unusual relationship. Inspired from [230] and [231], we focus on recognizing unusual actions performed by an actor in a video. To identify an unintentional action in the video, OOPs have investigated mid-level perceptual clues [232].

• **Federated learning-based action recognition:** Instead of centralizing all training data on a server or in the cloud, federated learning assigns the learning task to a

federation of participating devices that generate the training data. FedAvg [210] and FedMA [211] are a naive averaging approach and do not account for computational limitations in edge devices.

•**Motivation of the Chapter:** To the best of our knowledge, only one work has proposed federated learning algorithm for action recognition task [233]. The authors have proposed on asynchronous federated learning algorithm to remove the bottle neck problem. However, they have applied the knowledge of federated learning for habitual actions whereas in our work we are focusing on video-based unusual actions. The methods [162,225,225–227] only handles the habitual combination of human and object in a video. In contrast to recent methods, we handle unusual combination of human and object. Most of the above deep networks handles habitual action recognition (*i.e.,* run, jog) and ignore the learning representations of unusual actions like '**cutting keyboard**' and '**hammering laptop**'. By the aforementioned analysis, we have been inspired to create an architecture capable of incorporating an unusual combination of people and objects that results in unusual action.

5b.5 Proposed Approach

We tend to be interested in learning and identifying a patient's unusual action and behavior from a video in this work. Prior research considers habitual activities, such as drinking, eating, and holding [13, 37, 140]. Nevertheless, there is a lack of research on unusual actions that also exist in the real-world scenario, mainly among peoples suffering from mild cognitive disorders. Thus, we present a novel unusual behaviour network, named as (UnusualFedNet) to address the above mentioned issue. However, the disclose of privacy of any patient is another major issue in case of recognition of unusual actions. Motivated by the federated deep learning (FDL), we have incorporated the concept FDL on edge devices to rePrefacecognize unusual actions preserving the privacy concerns of the patients. Further, the task of recognizing an action is computa-

tionally heavy, but FDL solves this complexity issue too. It allows several edge devices to train a neural network model simultaneously under the supervision of a central server without sharing any data without swapping data samples manually. Our proposed network (UBNet) is further decompose into backbone module and attention module. The functional overview of UBNet is shown in Figure 5b.3. We first obtain spatiotemporal tensors utilizing off-the-shelf light-weight 3D feature extraction module. Next, we incorporate human-object relation module to get finegrained action information. Finally, we utilize federated learning algorithm to train the module locally on edge devices.

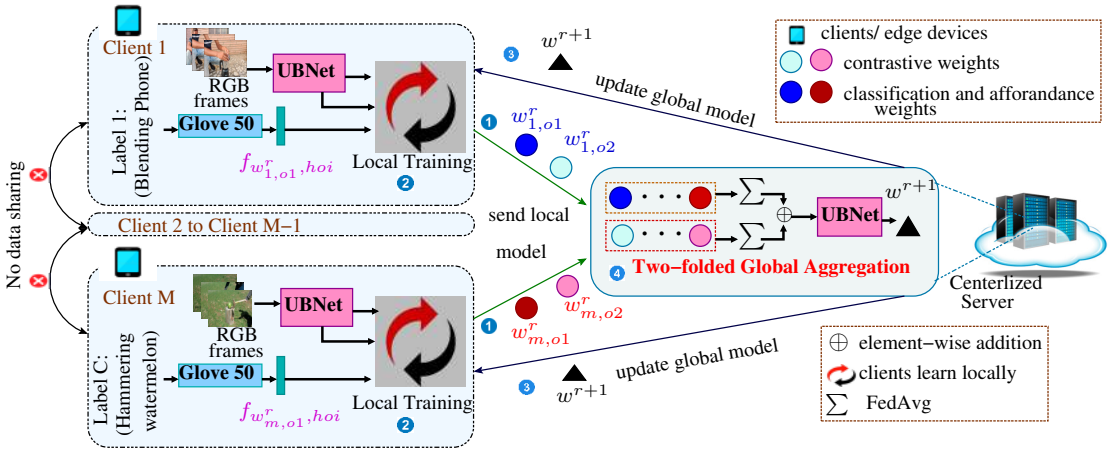


Figure 5b.2: Illustration of UnusualFedNet, a federated learning setup for unusual action recognition task. 1. Distributing data to edge devices 2. Each edge device run the UBNet (shown in Figure 5b.3) locally with chunk of dataset, shown in Figure 5b.5. 3. Each edge device send the updated weights to server. 4. Global weights are updated by server after each communication round.

5b.5.1 Feature Extraction Module: Light-weight ShuffleNet

In this section, we extract spatiotemporal cues from video sequences, which consist of appearance and motion features that assist in categorize the video in a particular action class. We consider popular 3D CNNs due to their exquisite impact on providing good contextual and action information. There are many variants of 3D CNNs, such as C3D, I3D, and R3D [129, 149, 234]. However, these variants are computationally heavy,

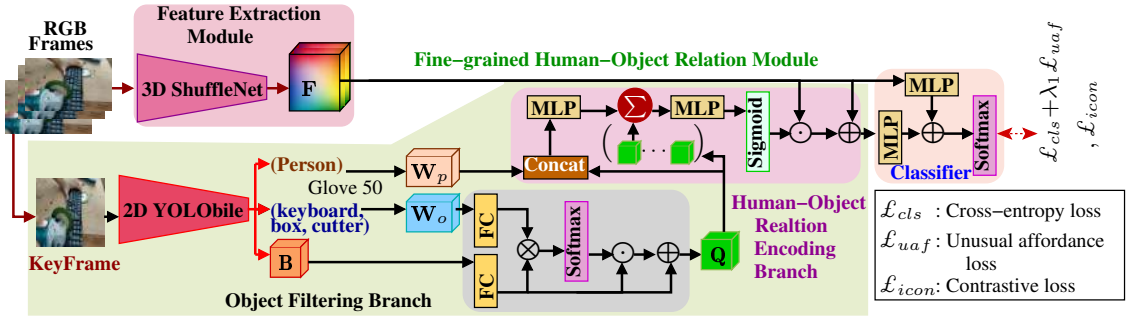


Figure 5b.3: Functional overview of UBNet. Here Feature Extraction Module depicts Light-weight ShuffleNet Architecture. $\otimes$ is element-wise multiplication, $\odot$ is hadamard product and $\oplus$ is element-wise addition operations.

making them difficult to deploy on edge devices. Therefore, we opt for light-weight CNNs, like ShuffleNet, MobileNet, and SqueezeNet, which are resource-efficient and resolve the deployment issue on edge devices [235–237]. The classification performance is maintained while the number of model parameters is reduced by these models. Here, we have chosen ShuffleNet [238] as our backbone module due to its robustness and lower training cost as compared with other models. It lay out the discriminative appearance and motion features of action performed by a person, while reducing the parameters almost to 0.55 millions as comapred with other CNN models [238].

We first fed a video sequence $\mathbf{V}$ with $\mathcal{T}$ RGB frames as an input and get the spatiotemporal feature tensor $\mathbf{F}$ as an output. The extracted feature tensor is of size $\mathcal{T} \times \mathbf{H} \times \mathbf{W} \times C$, where $\mathbf{H} \times \mathbf{W}$ is spatial resolution and C is feature channel dimension. Formally, extracted spatiotemporal features $\mathbf{F}$ is computed as: $\mathbf{F} = \mathbf{ShuffleNet}(\mathbf{V})$, where $\mathbf{ShuffleNet}(\cdot)$ is the collection of convolutional layers and channel shuffle blocks. The size of output tensor is $\mathcal{T}/16 \times \mathbf{H}/32 \times \mathbf{W}/32$ with $C = 960$. Finally, the resulted tensor is fed to the finegrained feature attention module to highlight the most fine features of an action present in the video.

5b.5.2 Fine-grained Human-Object Relation Module

In the case of video-based action recognition, all the cues, such as actors, objects, and their interrelations, play a vital role for capturing the interactive motion. However, there can be an issue of misclassification since many irrelevant cues which are not related to the action class may dominate. In other words, the relevant cues are mainly objects which are closely connected with humans and help to generalize the actions. For example, the vital pertinent elements for an action "**grinding a phone**" are the human, grinder, and phone, as shown in Figure 5b.4. Other irrelevant cues (such as background and boxes) should be discarded for predicting a precise action class. Distinct from recent HOI methods, which usually concentrate on the habitual interaction of humans and objects (e.g., "cutting fruits"), we handle the pair of human-object that rarely combine, as shown in Figure 5b.1 from second to fourth rows. The valid question arises is that how can a machine recognize the unusual relevant combination of human and object that infers to unusual action from a video? The problem of establishing the relationship between rare interactions between humans and objects motivates us to propose an fine-grained human-object relation module. To learn fine-grained, generalized represenation of unusual interaction and handle heterogeneity of local data distribution across clients, we incorporate novel affordance and instance contrastive loss, respectively. The key idea for unusual affordance loss is to get the accurate feature matching score between the word embedding of ground-truth and unusual action prediction vector. The instance contrastive loss helps in reducing the distance between the represenations of local and global model. Human-object relation module consists of two different branches, i.e., object filtering and human-object relation encoding. First, we extracted both the visual features and object class from the video keyframe using the lightweight pre-trained model. The keyframe is selected uniformly, *i.e.,* we have taken the middle frame from the set of frames as a keyframe, similar as [239]. We have utilized a pre-trained YOLObile [240] model on MSCOCO [241] dataset to extract visual

features $\mathbf{B}$ and objects from the keyframe of a video. Next, we have applied 50-dim GLOVE word embedding for each object to extract object features $\mathbf{W}$. Finally, we have divided the word features $\mathbf{W}$ into humans feature matrix $\mathbf{W_p}$ and object feature matrix $\mathbf{W_o}$.

Figure 5b.4: The video frames from UnusualAction dataset. There are multiple irrelevant elements that does not interact with a relevant actor. It shows '**grinding a phone**' action.

The possibility of noisy or irrelevant object prediction is very high because of utilizing a pre-trained model. The issue of noisy prediction motivates us to design the object filtering branch (OFB) to filter out the unneeded object that doesn't lead to unusual action. It takes the visual feature $\mathbf{B} \in \mathbb{R}^{n_b \times d_b}$ and object word predicate $\mathbf{W_o} \in \mathbb{R}^{n_w \times d_w}$ as an input to filter the irrelevant objects from a video frame. Both the feature matrices $\mathbf{B}$ and $\mathbf{W_o}$ contain number of instances $\{n_b, n_w\}$ and feature of each instance $\{d_b, d_w\}$. We have considered the object word feature $\mathbf{W_o}$ as the supervision information that helps to filter out the irrelevant objects in the visual feature matrix $\mathbf{B}$. To get the filtered information of objects, we first append fully-connected (FC) layers and then applied the softmax activation function to compute the weighted map Ψ and is formulated as follows:

$$\Psi = \mathbf{softmax}(\overline{\Psi}), \tag{5b.1}$$

$$\overline{\Psi} = \Upsilon_b(\mathbf{B})\Gamma_w(\mathbf{W_o}), \tag{5b.2}$$

where $\{\Upsilon_b(\cdot), \Gamma_w(\cdot)\}$ are the FC layers and produce the output of size $n_b \times d_{\overline{\Psi}}$ and $n_w \times d_{\overline{\Psi}}$, respectively. The weighted map Ψ is further multiply and added with visual

features to get the filtered object information $\mathbf{Q}$. The filtered matrix is calculated by Eq. (5b.3) as follows:

$$\mathbf{Q} = \overline{\mathbf{Q}} + \Upsilon_b(\mathbf{B}), \tag{5b.3}$$

$$\overline{\mathbf{Q}} = (\Psi \cdot \mathbf{1}^\top) \odot \Upsilon_b(\mathbf{B}), \tag{5b.4}$$

$$\Upsilon_b(\mathbf{B}) = (\mathfrak{W}_b \mathbf{b} + \mathsf{b}_b), \tag{5b.5}$$

where $\{\mathfrak{W}_b, \mathsf{b}_b\}$ are the learnable weights and biases of FC layer, $\mathbf{1}$ is the matrix whose elements are all equal to 1, and $\odot$ represents a hadamard product.

Usually in real world scenario, a person interact with surrounding objects by their limbs, for example, the person is putting the phone into the grinder using hands, as shown in Figure 5b.4. To maintain the relationship between the human and object, we need to estimate the accurate human-object interaction. Therefore, we devise the human-object relation encoding (HORE) branch to estimate the relation between human and filtered object. First, both the person and object features are concatenated to maintain the relationship between the human and each surrounding object. Then, we linearly projected the concatenated features of human and filtered objects to get the output of size $\mathbb{R}^{n_{p,o} \times d_q}$ in HORE. The projected feature map helps in inferring whether the object and human are interacting. Next, we have aggregated the relations with number of object instances n_o using another linearly projected layer. Finally, the relational score is calculated using sigmoid function, which is given by:

$$\mathbf{Z} = \mathbf{sigmoid}(\mathbf{MLP}(\sum_{k \in n_o} \mathbf{MLP}([\mathbf{W_p}, \mathbf{Q}], \mathbf{Q}_k))), \tag{5b.6}$$

where **sigmoid** is sigmoid function to constrain the elements $\mathbf{Z}$ to range in (0,1) and multi-layer perceptron **MLP** is the linearly projected layer that consist of one FC layer followed by ReLU and then FC layer. $[\cdot, \cdot]$ is the concatenation function. $\mathbf{Q}_k$ is the

filtered k object vector. Here, we primarily consider the relations which are valid for unusual human behaviour analysis. The relational score is used to enhance the interaction representation with element-wise multiplication. During experiment, we observe that adding the spatiotemporal feature $\mathbf{F}_t \in \mathbb{R}^{n_f \times d_f}$ in temporal dimension provides additionally improvement. The human-object interacted spatiotemporal feature tensor $\mathbf{H}$ is formulated by:

$$\mathbf{H} = \sum_{t \in \mathcal{T}/16} (\mathbf{Z} \odot \mathbf{F}_t) + \mathbf{F}_t. \tag{5b.7}$$

We concatenate the obtained human-object related attentive feature and spatiotemporal feature to obtained the overall interacted features. Finally, we fed the concatenated feature to the classifier for predicting the unusual action class. Formally, the above concept is mathematically represented by:

$$\mathbf{P} = \mathfrak{W}_f(\mathbf{F}) + \mathfrak{W}_h(\mathbf{H}), \tag{5b.8}$$

where $\{\mathfrak{W}_f, \mathfrak{W}_h\}$ are the trainable weights for spatiotemporal and temporalchannel attention branches. The trainable weights are obtained from **MLP** layers.

$$\mathbf{L} = \mathbf{softmax}(\mathbf{P}), \tag{5b.9}$$

where $\mathbf{L}$ is the predicted output obtained after softmax layer. We train the UBNet using cross entropy $\mathcal{L}_{cls}$ and unusual affordance $\mathcal{L}_{uaf}$ losses between the ground-truth $\mathbf{gt}$ and the prediction $\mathbf{L}$ obtained from UBNet. We freeze the weights obtained from cross entropy $\mathcal{L}_{cls}$ and unusual affordance $\mathcal{L}_{uaf}$ loss and then train UBNet using instance contrastive loss $\mathcal{L}_{icon}$, as shown in Figure 5b.5. We jointly minimize the local objective

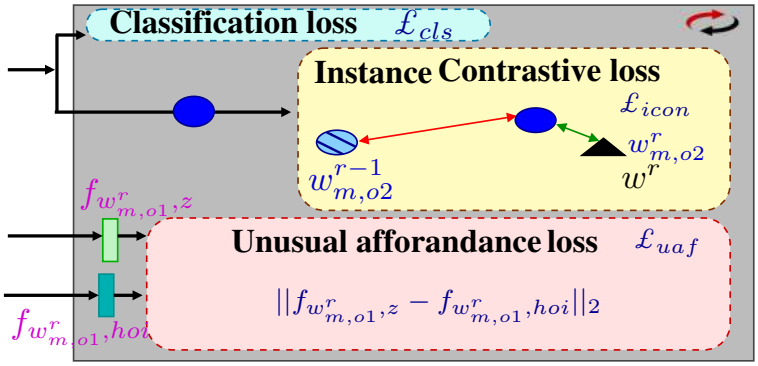


Figure 5b.5: Illustration of local training in UnusualFedNet. Red Arrow indicates minimize agreement between previous and current local feature vectors, whereas Green Arrow depicts maximize agreement between the current local feature vector and global feature vector.

functions of each client which are formulated by:

$$\mathcal{L}_{obj_1} = \mathcal{L}_{cls}(\mathsf{w}^r_{m,o1}; (\mathbf{gt}, \mathbf{L})) + \lambda_1 \mathcal{L}_{uaf}(\mathbf{f}_{\mathsf{w}^r_{m,o1},\mathbf{z}}; \mathbf{f}_{\mathsf{w}^r_{m,o1},hoi}), \tag{5b.10}$$

and

$$\mathcal{L}_{obj_2} = \mathcal{L}_{icon}(\mathsf{w}^r_{m,o2}; \mathsf{w}^{r-1}_{m,o2}; \mathsf{w}^r), \tag{5b.11}$$

where $\mathsf{w}^r_{m,o2}$ and $\mathsf{w}^{r-1}_{m,o2}$ are the current and previous local parameters of the model obtain from instance contrastive loss in each client m, respectively. w^r is global parameter. $\mathsf{w}^r_{m,o1}$ learnable parameter obtained from first objective function. λ_1 is the controlling hyper-parameters to control the weight of unusual affordance in each client. In unusual affordance loss, calculate the human-object interaction affordance matching score. We have projected word embedding of **gt** to the human-object interacted embedding space obtained in Eq. 5b.6. We have extracted word embedding $\mathbf{f}_{\mathsf{w}^r_{m,o1},hoi}$ using GLOVE50 in each client, given by:

$$\mathbf{f}_{\mathsf{w}^r_{m,o1},hoi} = GLOVE50(\mathbf{gt}). \tag{5b.12}$$

The obtain feature embedding vector is fed into sigmoid to get the scalar value indicating whether this unusual compositional human-object interaction is plausible. In instance

contrastive loss, first we have extracted the features of video from both local and global models using the Eq. 5b.8. Then, we have computed the similarity between the current, previous local feature vectors and global feature vector. Formally, the instance contrastive loss is explained as:

$$\pounds_{icon} = -\mathbf{log}\frac{\mathbf{h}(\mathbf{P}_{\mathsf{w}^r_{m,o2}}, \mathbf{P}_{\mathsf{w}^r})}{\mathbf{h}(\mathbf{P}_{\mathsf{w}^r_{m,o2}}, \mathbf{P}_{\mathsf{w}^r}) \ + \ \mathbf{h}(\mathbf{P}_{\mathsf{w}^r_{m,o2}}, \mathbf{P}_{\mathsf{w}^{r-1}_{m,o2}})}, \tag{5b.13}$$

where $h(\mathbf{u}, \mathbf{v}) = \mathbf{exp}(\frac{\mathbf{u}^\top \mathbf{v}}{||\mathbf{u}||_2||\mathbf{v}||_2}/\tau)$ and τ is temperature. $\mathbf{P}_{\mathsf{w}^r_{m,o2}}$ is current feature vector of local model and $\mathbf{P}_{\mathsf{w}^{r-1}_{m,o2}}$ is previous feature vector of local model. $\mathbf{P}_{\mathsf{w}^r}$ is feature vector of global model.

5b.5.3 Federated Contrastive Learning (FCL) setup

In this section, we consider the FCL setup with m edge devices where aim is to optimize the training loss across the m edge devices. Each m edge devices has different training datasets, *i.e.,* $\mathcal{D} = \{\mathcal{D}_1, \cdots, \mathcal{D}_m\}$. The dataset is further divided into training and testing sets on each m edge devices, formally given as $\mathcal{D}^m_{train}$ and $\mathcal{D}^m_{test}$, where $M \in \{1, 2, \cdots, m\}$. In FCL setup, all the edge devices utilize their own dataset to train the model locally and then communicate the local weights back to the centralized server. The server then aggregate the obtained weights and update the global model. Here, all edge devices train the same UBNet model (explained in Sections 5b.5.1 and 5b.5.2) with same training strategy. Moreover, no permission for data sharing between edge devices is given by server (as shown in the Figure 5b.2), *i.e,* data privacy has been maintained through out the local training. The whole training process is divided into three steps: (1) initialize parameters of UBNet on m edge devices, (2) local training on m edge devices in parallel mode, (3) global aggregation on centralized server. Illustration of our FCL setup with UBNet architecture is shown in the Figure 5b.2.

In the first step of training, we initialize the parameters, *i.e,* weights and biases of

Algorithm 5b.1: Federated Contrastive Unusual Action Recognition

Input: M clients, each having local datasets $\mathcal{D}_m$ with batch size B, number of communication rounds R, number of local epochs $\mathcal{E}$, hyperparameters λ_1, and learning rate η.;

Output: Final learnt UBNet parameters w_m on each clients after R rounds;

skip /**Initialization of FCL algorithm**/

skip **Server**():

Generate and randomly initialize the parameters of UBNet (w_0) at the server;

Server broadcasts UBNet to all the clients;

for *each round* $r \in R$ **do**

  Select random M clients;

  for *each client* $m \in M$ *in* ***parallel*** **do**

    $\mathsf{w}^r_{m,o1}, \mathsf{w}^r_{m,o2} \leftarrow$ **Local_Training**(r, m, w);

  /* *two-folded global aggregation* */

  $\mathsf{w}^{r+1}_{m,o1} = \sum_{m \in M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathsf{w}^r_{m,o1}$; $\mathsf{w}^{r+1}_{m,o2} = \sum_{m \in M} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} \mathsf{w}^r_{m,o2}$; $\mathsf{w}^{r+1} = \mathsf{w}^{r+1}_{m,o1} + \mathsf{w}^{r+1}_{m,o2}$;

return w^R;

Local_Training(r, m, w) :

/**Run on client* m*/

$\mathsf{w}^r_{m,o1}, \mathsf{w}^r_{m,o2} \leftarrow split(\mathsf{w}^r)$;

for *each* $e \in \mathcal{E}$ **do**

  for *each* $B \leftarrow$ ***random_split***$(\mathcal{D}_m)$ **do**

    $\mathcal{L}_{cls}(\mathsf{w}^r_{m,o1}) = CrossEntropyLoss(\mathbf{gt}, \mathbf{L})$;

    /* $||\cdot||_1$ is vectorized **L1** loss*/

    $\mathcal{L}_{uaf} = ||\mathbf{f}_{\mathsf{w}^r_{m,o1,\mathbf{z}}} - \mathbf{f}_{\mathsf{w}^r_{m,o1,hoi}}||_1$;

    $\mathcal{L}_{obj_1} = \mathcal{L}_{cls}(\mathsf{w}^r_{m,o1}) + \lambda_1 \mathcal{L}_{uaf}(\mathbf{f}_{\mathsf{w}^r_{m,o1},\mathbf{z}}; \mathbf{f}_{\mathsf{w}^r_{m,o1,hoi}})$,

    $\mathsf{w}^r_{m,o1} \leftarrow \mathsf{w}^r_{m,o1} \nabla \mathcal{L}_{obj_1}$;

    /* *Here,* $\mathbf{P}$ *is mapped represenation obtained in Eq. 5b.8,*τ *is temperature parameter, and* $||\cdot||_2$ *is* ***L2*** *norm.* */

    $\mathcal{L}_{icon} = -\mathbf{log} \frac{\mathbf{h}(\mathbf{P}_{\mathsf{w}^r_{m,o2}}, \mathbf{P}_{\mathsf{w}^r})}{\mathbf{h}(\mathbf{P}_{\mathsf{w}^r_{m,o2}}, \mathbf{P}_{\mathsf{w}^r}) + \mathbf{h}(\mathbf{P}_{\mathsf{w}^r_{m,o2}}, \mathbf{P}_{\mathsf{w}^{r-1}_{m,o2}})}$;

    $h(\mathbf{u}, \mathbf{v}) = \mathbf{exp}(\frac{\mathbf{u}^\top \mathbf{v}}{||\mathbf{u}||_2 ||\mathbf{v}||_2} / \tau)$;

    $\mathcal{L}_{obj_2} = \mathcal{L}_{icon}(\mathsf{w}^r_{m,o2}; \mathsf{w}^{r-1}_{m,o2}; \mathsf{w}^r)$;

    $\mathsf{w}^r_{m,o2} \leftarrow \mathsf{w}^r_{m,o2} \nabla \mathcal{L}_{obj_2}$;

return $\mathsf{w}^r_{m,o1}$ and $\mathsf{w}^r_{m,o2}$ to server;

UBNet model on all m edge devices. Then, we randomly select set of m edge devices for each communication round r. Next, we update the weights of model for each edge device m that run in parallel mode. Each edge device m trains its local UBNet model with its own dataset $\mathcal{D}_{train}^{M}$ and $\mathcal{D}_{test}^{M}$ for $\mathcal{E}$ epochs individually. After local training, all edge devices upload their local parameters to the global centralized server. For two-folded global aggregation step, we adopt the federated averaging algorithm FedAvg [242] and average all local parameters to obtain the global parameters. Then, the latest global parameter is returned to all edge devices and the global server obtains the final global parameters after R communication rounds. Algorithm 5b.1 explains the entire process of training in mathematical formation.

5b.6 Experimental Results

In this section, we validate the robustness of UBNet through the empirical analysis on RareAct [224] and UnusualAction datasets. We conduct the ablation study to assess how well our architecture works. These tests are run on centralized servers and edge devices equipped with i7-6800K CPUs and NVIDIA Geforce GTX 1080Ti GPUs. To show that our FCL optimization is resilient to heterogeneous edge devices, such as the NVIDIA Jetson AGX Xavier with 32GB memory and 512-core Volta GPU and the NVIDIA Jetson Xavier NX with 8GB memory, we used a range of clients. Using the Pytorch platform, the models are put into practice. For identifying actions, we have used the performance metric accuracy (Acc.).

We have conducted experiment on RareAct, a benchmark dataset for rare action detection based on videos that is available to the general audience. Since we only have one dataset that focuses solely on rare action, we have also included the dataset with at most 20% overlap in rare action to show rare verb-noun and action classes. We also compiled videos of unusual activity under the term “UnusualAction” dataset and reported the experimental results.

5b.6.1 Publicly available Dataset

RareAct. This dataset consist of unusual actions with unlikely compositions of common action verbs and object nouns. It consists of 122 different action classes through combining verbs and nouns rarely co-occurred together. The number of videos provided are 905. RareAct includes 7603 rare videos clip instances with length of 10 seconds. Each action class is split into positive, negative, and hard negative video instances. There are 1765 positive, 19 verb, and 38 noun samples.

5b.6.2 Our UnusualAction Dataset

Motivation. Many datasets for habitual action recognition has been presented in recent years with rich annotations. However, research is still mainly limited to unusual human action recognition. Thus leaving a significant gap towards describing the unusual action of a video. We fill this gap by presenting a new "**UnusualAction** dataset", in short UAD. It is organized hierarchically in a semantic taxonomy that focuses on video-based human-object interactions related to unusual actions.

Data collection, annotation, and validation. The dataset was collected by 32 individuals, where 20 are males and 12 are females. Participants are asked to collect the videos that have rare and unusual actions. Basically, the video for each class are obtained by first searching on YouTube or Movies. Then participants are asked to decide if the clip contains the accurate unusual action or not. Five or more confirmations (out of ten) are required before a clip was accepted. Moreover, we asked participants to annotate the clip-level based on whether the human made hand contact with an instance of 10 object categories, such as drill, shoes, saw, knife, grinder, mobile, micro-oven, hammer, watermelon, and laptop.

UAD contains approximate 1K videos in total with 14 action classes for training and validation set. We split the UAD into $\{75\%, 15\%, 10\%\}$ for training, validation and testing, respectively. The classes are '**Blending phone**', '**Crushing laptop**', '**Cutting**

keyboard', '**Drilling laptop**', '**Drilling phone**', '**Frying Phone**', '**Hammering laptop**', '**Hammering phone**', '**Hammering pumpkin**', '**Hammering watermelon**', '**Microwave shoes**', '**Microwave phone**', '**Washing phone**', and '**Washing laptop**'. UAD encompasses semantic aspects defined on categories of objects, actions, noun, and verb, which naturally captures the real-world scenarios. All types of variation are included in our dataset, i.e., variations in camera motion, size of objects, background, illumination, and viewpoints. The minimum and maximum resolution of a frame is 480 and 1080, respectively. We will freely release this data along with all annotations and split information to the community.

5b.6.3 Implementation Details

In this section, we describe the details related to input, data augmentation, feature extraction module, optimization parameters, and inference related attributes.

Data augmentation. Before providing the video frames as an input to the lightweight ShuffleNet backbone network, we scale each frame to a spatial resolution of 224×224. On both datasets, we have downsampled the videos at 15 frames per second for training. Each dataset is chosen to have a temporal length of 32 distinct frames for action video. If the videos aren't long enough, we have padded the zeros. We adjusted the training batch size for each edge device in the FCL setup to 8 in order to fit each batch into the GPU memory.

Feature Extraction Module. We have initialized parameters of ShuffleNet via training on the Kinetics dataset [133]. Because the obtained features have larger receptive fields and more contextual data, we use the ShuffleNet network. The value of $\mathcal{T}$ is taken 32 in training as well as testing with spatial size of 224×224. Dropout is appended after the every convolutional layer of backbone module with a rate of 0.3.

Optimization parameters. The initial parameters are set up with the help of weights from ShuffleNet and we use Adam optimizer to optimize the our UBNet. β_1 and β_2 are

set to 0.9, respectively and the value of ϵ is set to 10^{-5}. Experiments show that 80 epochs are enough to converge the model locally in each edge device. The communication rounds are 50 for global aggregation of weights.

Inference. When compared to other state-of-the-art methods, we employ the standard procedures outlined in the literature for inference time. For the purpose of predicting the action class, a 32-frame video clip is divided into 10 sampled segments.

5b.6.4 Ablation Study and Qualitative Analysis

We test our architecture on the UnusualAction and RareAct datasets in this section. To demonstrate efficacy, we present ablation studies on various aspects. In addition, we demonstrate the visual results for both routine and unusual actions.

Impact of backbone networks. In addition to analysis of different components, we have experimented with different backbone modules such as MobileNet, ShuffleNet, and SqueezeNet. We have reported the FLOPs of existing methods in Table 5b.1 to compare the complexity for selecting backbone network. We have retrained the existing methods for fair comparison, taken from [16]. In this ablation study, we have only consider backbone network and neglect the attention part for proper selection of backbone network. We observe that FLOPs of 3DShuffleNetV1 0.5x is less than other backbone networks, although the accuracy is little less. We also observe that 3DShuffleNet V2 1.5x has better accuracy for both the datasets. Thus, we have incorporated 3DShuffleNet V2 1.5x as our backbone module for better feature extraction.

Consequence and analysis of individual components of UBNet. To understand how each component in UBNet contributes to overall accuracy, we examine the impact of each component. Table 5b.2 presents the findings in terms of accuracy. The results of ShuffleNet, our baseline for the UnusualAction and RareAct datasets, are reported in the top row of Table. Even before to reporting the results, OFB is first added in parallel with the ShuffleNet network. After that, OFB and HORE are combined to use

Table 5b.1: Comparison of computational complexity with SOTA methods for UnusualAction and RareAct datasets. * shows that we have retrained the models on our dataset.

Backbone	**RareAct Acc.**	**UAD Acc.**	**FLOPs**	**params**
3DShuffleNet V2 1.5x*	**60.7**	**63.5**	291M	3.16M
SqueezeNet*	60.4	63.1	926M	2.15M
3DMobileNet V2 0.7x*	60.6	63.4	325M	2.05M
3DShuffleNet V1 1.0x*	60.5	63.3	199M	1.52M
3DMobileNet V2 0.42x*	59.3	61.2	177M	1.40M
3DMobileNet V1 0.5x*	59.6	61.7	98M	1.17M
3DShuffleNet V2 0.25x*	59.7	62.8	116M	0.83M
3DShuffleNetV1 0.5x	60.6	63.2	**78M**	**0.55M**

ShuffleNet to depict the outcomes. As an illustration, "ShuffleNet + OFB" is one of the combinations to explain the outcomes in comparison to other combinations. Due to the fact that it provides more contextual and pertinent information about an action, we can see that our network performs better than a single branch. Additionally, we also note that each component is essential for improving the baseline performance although the computational complexity is an overhead.

Table 5b.2: Impact on computational complexity of UBNet for UnusualAction and RareAct datasets.

Models	**RareAct Acc.**	**UAD Acc.**	**params**	**FLOPs**
ShuffleNet	60.6	63.2	**0.55M**	**78M**
ShuffleNet + OFB	60.8	63.5	0.75M	78.5M
ShuffleNet + HORE	61.0	63.8	0.65M	78.3M
UBNet	**65.9**	**70.4**	0.90M	79.5M

Impact of fusion strategy. The comparison results for various fusion strategies are shown in the Table 5b.3. Our results suggest that UBNet extracts fine cues with greater precision. We demonstrate that the performance of unusual action recognition is improved by the weighted concatenation fusion technique, which outperforms the other existing techniques. Additionally, we draw the conclusion that concatenation

alone is insufficient to provide action-specific discriminatory fine-grained information. As a result , we have chosen the weighted concatenation fusion strategy.

Table 5b.3: Performance of fusion strategy of our architecture on UnusualAction and RareAct datasets.

Stages	**RareAct Acc.**	**UAD Acc.**
element-wise product	58.7	60.6
weighted element-wise product	59.0	61.4
stacking	59.6	61.9
weighted stacking	60.8	65.7
concatenate	64.5	69.4
weighted concatenate	**65.9**	**70.4**

FCL setup on different distributions of dataset and aggregation technique. We setup the FCL experiments on two different distributions of training dataset, which we marked as $\mathcal{D}_{iid}$ and $\mathcal{D}_{non-iid}$ with single and two-folded aggregation techniques. The $\mathcal{D}_{iid}$ depicts that training dataset is equally divided to each edge devices and known as identically independent distribution (iid) whereas $\mathcal{D}_{non-iid}$ depicts that data is distributed in unequal fashion. Table 5b.4 summarizes the details of data given to each edge devices with respect to iid or non-iid and report the accuracy for both the cases, including single and two-folded aggregation. We observe that the results of iid with two-folded aggregation outperforms non-iid strategy due to equal distribution of data. Moreover, we also observe that the accuracy drop in case of non-iid is occur due to poor generalization between the local and global model for both aggregation techniques.

Accuracy of UBNet at FCL setup with different number of edge devices and data distribution. With the help of an FCL setup, we have illustrated the effectiveness of our proposed network. Figure 5b.6 shows the effect of number of edge devices require to analysis the performance of UBNet. Moreover, we have also taken two cases of dataset distribution: a) iid and b) non-iid to observe the behaviour of

Table 5b.4: Performance of iid and non-iid FCL setup on UnusualAction and RareAct datasets.

FCL setup	Aggregation technique	RareAct Acc.	UAD Acc.
$\mathcal{D}_{iid}$	Single	63.8	68.6
	Two-folded	**65.9**	**70.4**
$\mathcal{D}_{non-iid}$	Single	52.6	60.3
	Two-folded	55.3	62.1

number of edge devices with respect to data distribution. We observe that after 6 edge devices the accuracy drops down in both the case of data distribution. Thus, we have selected 6 edge devices in FCL setup to reproduce better results.

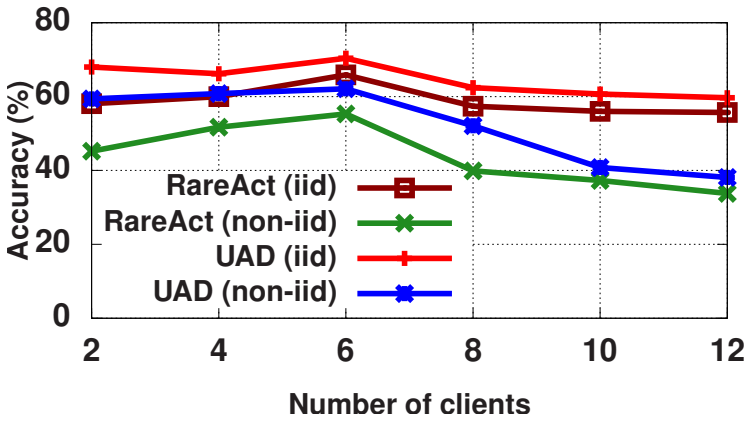
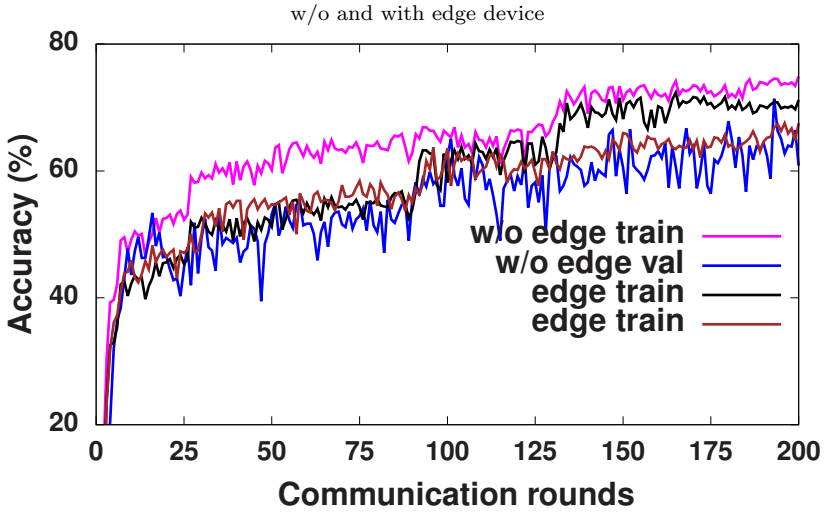
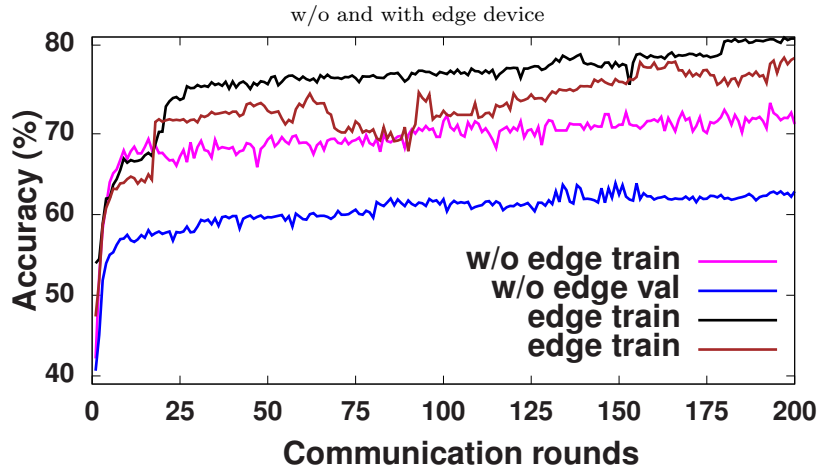


Figure 5b.6: Impact of number of edge devices (clients) on different distribution of data on RareAct and UAD datasets.

Accuracy of UBNet with FCL setup with and without the edge devices. We perform the experiment of our network with and without the number of edge devices and report the accuracy on datasets. Here, without the edge devices setup is termed as centralized training. As shown in the Figure 5b.7, UBNet achieves accuracy of 71.2% on RareAct and 77.2% on UAD dataset without the setup of any edge devices. The results with respect to 6 edge devices are 65.9% and 70.4% on RareAct and UAD, respectively. In our experiment, we consider iid dataset distribution for fair comparison on both the datasets. We observe that centralized results are always better than federated results.



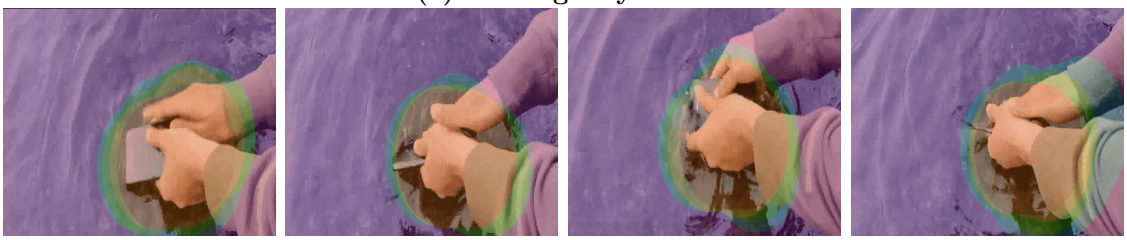
(a) RareAct dataset.



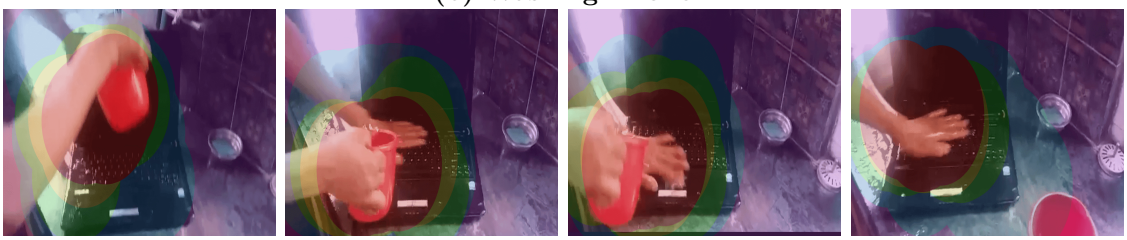
(b) UAD dataset.

Figure 5b.7: Illustration of performance of datasets on the convergence rounds with and without edge device in terms of accuracy.

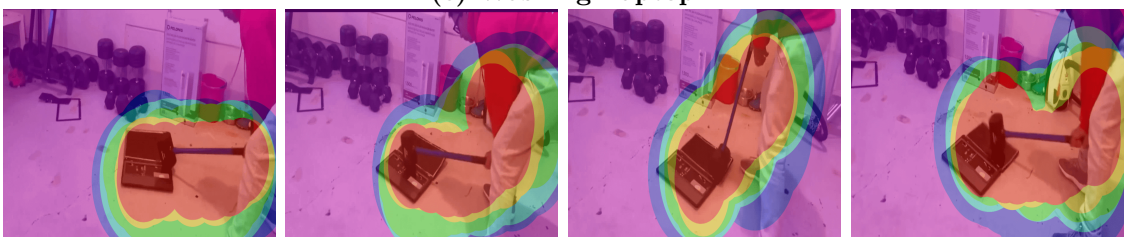
(a) Cutting Keyboard



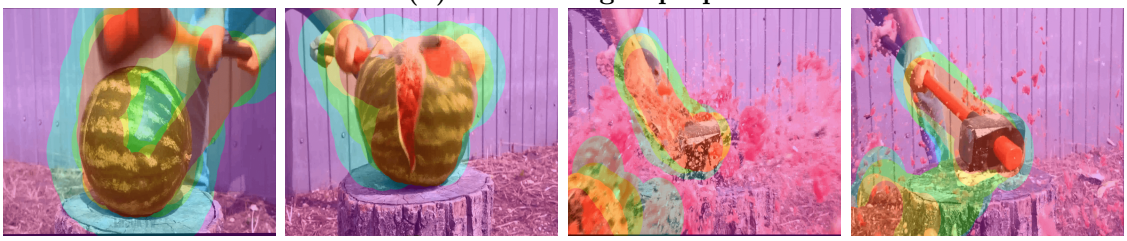
(b) Washing Phone



(c) Washing Laptop



(d) Hammering Laptop



(e) Hammering Watermelon

Figure 5b.8: Visualization of activation maps on RGB frames taken from UAD for **unusual action** classes with CAM.

Effect of controlling hyper-parameter λ. Empirically, we set the different value of λ to observe the variation of unusual affordance loss functions. We reported our results in the Table 5b.5 on the UnusualAction and RareAct dataset in terms of accuracy. In our experiment, we vary the values of λ_1 between the range of 0.3 and 0.9 for RareAct and set 0.4 to 1.0 for UnusualAction dataset. In case of UnusualAction dataset, we find out that for 0.3 the accuracy drops to 60.4 %. As the accuracy is very low, therefore we neglect the result in the Table 5b.5. We observe that when the value of λ_1 is 0.5, our model outperforms other models in case of RareAct. Similarly, we reported the results on UnusualAction dataset with same values of λ_1 under the same iid setting in Table 5b.5. The best result is at the value of 0.7. This also helps us to understand that our overall model is effect and robust with respect to the both heterogeneity and reasoning of unusual action. We also observe that as the value of λ_1 increases the performance degrades as model incline more towards heterogeneity with both the datasets.

Table 5b.5: Effectiveness of λ in the local objective function on RareAct and UnusualAction dataset.

RareAct							
λ_1	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Acc.	61.5	62.3	**65.9**	59.3	58.4	58.0	57.2
UnusualAction							
λ_1	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Acc.	65.8	66.0	67.3	**70.4**	65.0	64.6	64.1

Effect of loss functions $\mathcal{L}_{cls}$, $\mathcal{L}_{icon}$, and $\mathcal{L}_{uaf}$. We compare the effect of proposed loss functions with the help of multiple different combinations of losses in FCL setup to limit the generalization and heterogeneity. We reported our results in the Table 5b.6 on the UnusualAction and RareAct dataset in terms of accuracy. We observe that only using individual loss function is not sufficient to improve the accuracy as compared to other combination of loss on RareAct and UnusualAction dataset. Moreover, we also observe that the results obtain from the combination of classification $\mathcal{L}_{cls}$ and

Table 5b.6: Effectiveness of $\pounds_{cls}$, $\pounds_{icon}$, and $\pounds_{uaf}$ on RareAct and UnusualAction dataset.

Loss functions	RareAct	UnusualAction
$\pounds_{cls}$	56.1	63.6
$\pounds_{icon}$	57.9	62.8
$\pounds_{uaf}$	55.3	61.6
$\pounds_{cls} + \pounds_{uaf}$	60.6	65.7
$\pounds_{cls} + \pounds_{icon}$	59.7	64.3
$\pounds_{uaf} + \pounds_{icon}$	55.0	61.2
$\pounds_{cls} + \pounds_{uaf} + \pounds_{icon}$ (Ours)	**65.9**	**70.4**

unusual affordance loss $\pounds_{uaf}$ is better than the other combination of loss functions. As shown in the Table 5b.6, our loss functions consistenly produces the best results on the both datasets, which indicate the effectiveness of the combining the loss functions in increasing the performance of network.

Heat map results on UnusualAction dataset. We visualize the efficacy of our UBNet and show the results in class activation map (CAM) in Figure 5b.8. CAM coating over some RGB frames of each input video segment is shown in the Figure 5b.8. We notice that CAM covers large area on objects and/or hands when they appear, whether or not some action is taking place. As in the second example action "Washing mobile phone" the rate of change of motion is very slow and most of attention is on hand and mobile.

5b.6.5 Results on Habitual Actions Dataset

For the fair comparison, we consider usual action datasets, which are publicly available without a federated learning setup, to check the effectiveness of UBNet on usual dataset. In addition, we select those datasets that consist of noun-verb combinations in our experiments. First, we replace the size of channels in the classifier module present in UBNet according to the number of classes present in the dataset. Then, train the UBNet with the same values of hyperparameters and Adam optimizer for all experiments on

the usual action dataset.

Details of Publicly-available Usual Dataset The details of publicly available datasets are given as follows:

EPIC-Kitchens (EpK) [243]. It comprises of 90,000 egocentric movies shot in kitchens and lasted for a total of 100 hours. Each video has a "noun" and a "verb" labeled on it. The primary metric for this dataset is the accuracy, which are commonly mentioned. The highest scoring noun and verb pair is chosen to represent "activity".

Something-Something V2 (SSV2) [134]. The dataset comprises of 174 fine-grained actions related to HOI. It contains 220K videos in total with average length of 4.0 seconds. Each video is represents only one category of focus activity.

Comparison with State-Of-The-Art (SOTA) Results We compare our method with a number of SOTA action models in Tables 5b.7, 5b.8, 5b.9 and 5b.10 . The action recognition objective for EpK dataset consists of three tasks: classifying the verb, noun, and action in a video. We adopt the experiment setting and metrics as given in [244] to predict noun, verb, and action class. We first replace the classification layer of UBNet with two output fully-connected layers, one for verbs and one for nouns. The updated model is trained with an averaged softmax cross-entropy loss over each classification layer. The Top-1/5 accuracy is reported in Table 5b.7. UBNet report 58.2%, 28.3% Top-1 accuracy to predict verb and action, respectively, which makes 0.2% and 2.1% absolute improvements over TSN, TRN, and TRN-Multiscale. However, the prediction of noun is slightly low as compare to TSN model in case of centeralized setup. We also report the results in FCL setup to check the accuracy of our UBNet in Table 5b.8. The performance of TSN, TRN, and TRN-Multiscale are still less than our network for verb and action prediction in terms of Top-1 accuracy.

We show the results of the 2D based methods TSN, TRN, TRN-Multi-scale, and TSM to compare with UBNet with the BN-Inception as a backbone. The results on the validation set and test set of SSV2 dataset are listed in Table 5b.9. We observe that

the proposed UBNet outperform other models in both centralized and federated setup. The improvement performance gain is 1.5%/0.4% in terms of Top-1/5 test accuracy for centralized setup and 1.0%/0.7% in terms of Top-1/5 test accuracy for federated setup.

Table 5b.7: Comparison with SOTA on EPIC-Kitchens with 8 frames per video and 10 crops in test and only RGB frames with backbone BN-Inception (centralized).

Models	**Verb**		**Noun**		**Action**	
	Top-1	**Top-5**	**Top-1**	**Top-5**	**Top-1**	**Top-5**
TSN	47.5	87.0	**38.0**	**65.1**	22.4	44.1
TRN	58.0	**87.1**	36.2	63.0	25.4	45.6
TRN-Multiscale	57.2	86.4	37.4	63.2	26.2	46.1
UBNet	**58.2**	**87.1**	37.8	65.0	**28.3**	**47.6**

Table 5b.8: Comparison with SOTA on EPIC-Kitchens with 8 frames per video and 10 crops in test and only RGB frames (federated).

Models	**Verb**		**Noun**		**Action**	
	Top-1	**Top-5**	**Top-1**	**Top-5**	**Top-1**	**Top-5**
TSN	45.1	85.2	**36.7**	**64.3**	20.1	40.3
TRN	**55.7**	**84.2**	34.8	60.7	21.3	42.7
TRN-Multiscale	54.1	83.4	35.3	60.1	24.7	44.8
UBNet	**55.7**	84.0	35.0	58.4	**25.1**	**45.0**

Table 5b.9: Comparison with SOTA on Something-Something V2 with 8 frames per video and 10 crops in test and only RGB frames (centralized).

Models	**Val**		**Test**	
	Top-1	**Top-5**	**Top-1**	**Top-5**
TSN	30.0	60.5	-	-
TRN	55.5	83.0	56.2	83.1
TRN-Multiscale	48.8	77.6	50.8	79.3
TSM	59.1	85.6	-	-
UBNet	**59.3**	**86.1**	**57.9**	**83.5**

5b.7 Conclusion of the Chapter

In this work, we make the first attempt to explore the problem of recognizing the unusual action recognition in videos. Specifically, we propose a novel fine-grained human-object

Table 5b.10: Comparison with SOTA on Something-Something V2 with 8 frames per video and 10 crops in test and only RGB frames (federated).

Models	Val		Test	
	Top-1	**Top-5**	**Top-1**	**Top-5**
TSN	25.2	54.2	-	-
TRN	52.8	80.1	54.7	80.1
TRN-Multiscale	42.6	74.3	48.3	75.3
TSM	57.1	83.7	-	-
UBNet	**57.7**	**84.2**	**55.7**	**79.4**

relation module to learn the relevant unusual relationship between human and object information and increase the performance of proposed architecture. Moreover, we enable the computationally heavy action recognition task on the clients that have less memory storage and limited computing power. For the first time, federated learning is incorporated for the video-based unusual action recognition task. The proposed solution trains the recognition model in a decentralized manner that preserve privacy and reduce data communications overload. Utilization of FedAvg algorithm, we demonstrate the proposed FL framework for the unusual action recognition task. We also propose the affordance loss to identify the unusual human-object interaction more accurately. To maintain the heterogeneity of different client, we incorporate the instance contrastive loss. We validate the proposed method with both unusual and habitual datasets for fair comparison. Experiments conducted on unusual daasets, like RareAct and UnusualAction. We also perform experimetation on habitual action dataset, such as EpK and SSV2, and our network achieves the superior accuracy on EpK and SSV2 dataset.

5b.8 Publication related to the Chapter

1. Nitika Nigam and Tanima Dutta, "Finegrained Action Recognition for Unusual Human Behaviour Probe via Federated Contrastive Learning", in process.