http://www.scirp.org/journal/jbise

J. Biomedical Science and Engineering,
2019, Vol. 12, (No. 2), pp: 183-196

Applying Deep Learning Models to Mouse Behavior Recognition

Ngoc Giang Nguyen[1], **Dau Phan**[1], **Favorisen Rosyking Lumbanraja**[1], **Mohammad Reza Faisal**[1], **Bahriddin Abapihi**[1], **Bedy Purnama**[1], **Mera Kartika Delimayanti**[1], **Kunti Robiatul Mahmudah**[1], **Mamoru Kubo**[2], **Kenji Satou**[2]

[1]Graduate School of Natural Science and Technology, Kanazawa University, Kanazawa, Japan; [2]Institute of Science and Engineering, Kanazawa University, Kanazawa, Japan

Correspondence to: Ngoc Giang Nguyen, giangnn.bkace@gmail.com

Keywords: Mouse Behavior Recognition, Deep Learning, I3D Models, R(2 + 1)D Models

Received: November 28, 2018 **Accepted:** February 25, 2019 **Published:** February 28, 2019

ABSTRACT

In many animal-related studies, a high-performance animal behavior recognition system can help researchers reduce or get rid of the limitation of human assessments and make the experiments easier to reproduce. Recently, although deep learning models are holding state-of-the-art performances in human action recognition tasks, these models are not well-studied in applying to animal behavior recognition tasks. One reason is the lack of extensive datasets which are required to train these deep models for good performances. In this research, we investigated two current state-of-the-art deep learning models in human action recognition tasks, the I3D model and the R(2 + 1)D model, in solving a mouse behavior recognition task. We compared their performances with other models from previous researches and the results showed that the deep learning models that pre-trained using human action datasets then fine-tuned using the mouse behavior dataset can outperform other models from previous researches. It also shows promises of applying these deep learning models to other animal behavior recognition tasks without any significant modification in the models' architecture, all we need to do is collecting proper datasets for the tasks and fine-tuning the pre-trained models using the collected data.

1. INTRODUCTION

Researchers widely use many animals from fruit flies, mice to primates for studying biology, psychology or for developing new therapies or medicines. In many researches, observing the behaviors of the animals is a crucial step to get the data which is needed for answering research questions. Since watching and annotating the behaviors of these animals in hours of video clips are hard works, it's necessary to have a reliable and automated behavior recognition system to delegate these works to computers. With a well-performed system, we could not only solve the problem of the limitation of human assessments but also make the experiments easier to reproduce.

Many studies reported works in creating such systems for animal behavior recognition tasks. In the paper of Jhuang H. *et al.* [1], they created a system to automatically analyze behaviors of mice in home-cages. The system contains two modules: a feature extraction module and a classification module. In the feature extraction module, for each frame, they calculated the mouse's position and velocity based features and combined them with motion features which are extracted from the adjacent frames using an algorithm in [2]. These features then fed into an SVMHMM (Support Vector Machine-Hidden Markov Models [3]) to assess the action in the frame. In another research [4], Jiang J. *et al.* also used a similar approach but with a different feature extractor and classifier. For the feature extraction module, they detected interest points using a modified version of the algorithm in [5], then they extracted contextual and visual features from these points. And they fed these extracted features into a shallow neural network that has only one hidden layer to assess the actions in the frames. The changes in the feature extraction method and the classification algorithm slightly improved the performance of the system in comparison to the previous paper. And it showed that the design of the feature extraction module can affect the performance of the whole system. However, creating good feature extractors is not an easy task. It requires much expert knowledge and carefulness and it is not always successful. And the abilities of these created systems are highly limited to the problems they were designed to solve. For example, an automated mouse behavior recognition system may not work well in a raccoon behavior recognition task, although the two animals are sharing many similarities in their appearance.

We could solve the above problem by using deep learning models which have the ability of automated learning to extract useful features from given data. Because of having this ability, deep learning models are widely used in many application fields from computer vision, speech recognition to natural language processing and often become state-of-the-art models in the field they applied to. And we can use the same models for tasks that have similarities without significant changes in the architecture of the models.

Though its high performance, it is not easy to apply deep learning models for whatever tasks we have because these models have too many parameters that it requires an extensive amount of data to train these parameters. And it is one of the reasons why deep learning models have very high performances in human action recognition tasks but not well-studied in applying to animal behavior recognition tasks.

To fill in this gap, in this research, we investigated two current state-of-the-art human action recognition deep learning models in applying to a mouse behavior recognition task. The first model we investigated is the I3D model [6], which implements an inflated version of the inception module architecture [7]. The most important features of inception module are the utilization of the combined effects of filters of different sizes and pooling kernels all in one layer; and the usage of 1×1 convolutional filters which not only help to reduce the number of parameters but also introduce new combinations of features to its next layers. The second model we investigated in this research is the R(2 + 1)D model [8] which implements a 3D version of the residual module architecture [9]. This architecture allows the model to go deeper by solving the vanishing of information when training deep models.

To deal with the scarcity of training data, we did not train the models from randomly-initialized parameters but we used the parameters that were pre-trained on human action recognition tasks. By doing so, we can transfer knowledge that related to action recognition from human's tasks to the new models [10].

In the next section, we show the dataset which we used to evaluate the performances of the two deep learning models in the mouse behavior recognition task. In Section 3, we describe in detail experiments and results of the evaluating process. Finally, we give some conclusions in Section 4.

2. THE MOUSE BEHAVIOR DATASET

In the research of H. Jhuang, *et al.* [9], they introduced a task of neurobehavioral analysis of mouse phenotypes by monitoring the mouse's behaviors over long periods of time. In this experiment, each mouse is put in a transparent home cage, and there behaviors are recorded from a perpendicular angle to the side of the cages using consumer grade cameras.

In order to create a machine learning system to automatically analyze mouse's behaviors, Jhuang and his colleges have created a mouse behavior dataset by annotating the mouse's behaviors in over 10 hours of recorded videos. In their dataset, they have annotated 8 types of behavior: drinking, eating, grooming, hanging, rearing, walking, resting and micro-movements of the head. Example scenes of these behaviors are shown in Figure 1, and descriptions of these behaviors are shown in Table 1.

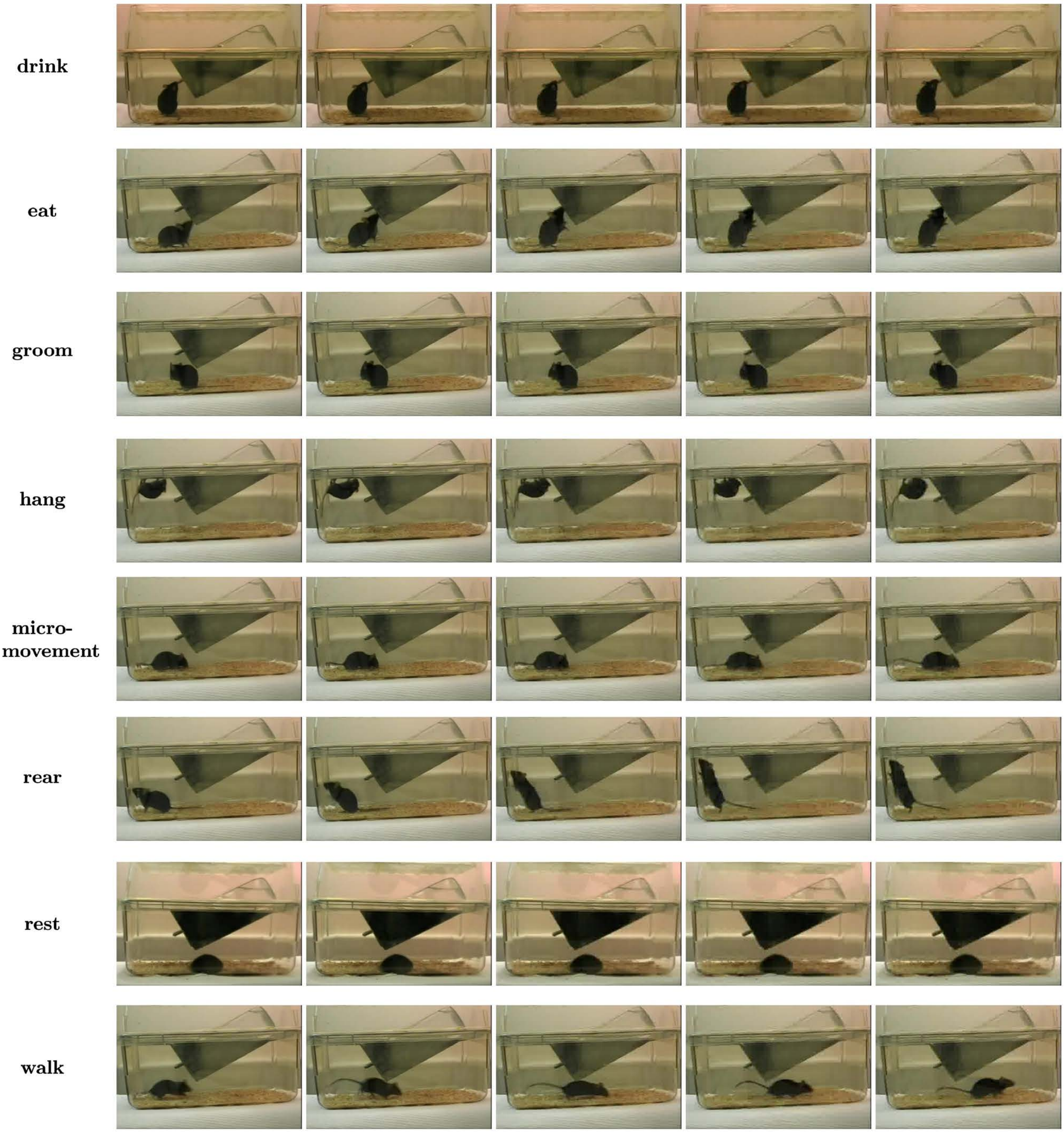


Figure 1. Example scenes of mouse's bahaviors.

In the created mouse behavior dataset, among totally more than 9000 short clips, only 4200 clips that are most unambiguous were selected to create the "clipped database". It includes about 285,000 frames and corresponds to about 2.5 hours of recorded videos. In this research, in order to properly evaluate the performance of the deep learning models, we decided to use only this subset to eliminate the ambiguous in the data that even human cannot declare. The distribution of number of frames of each behavior in the "clipped database" is shown in Figure 2.

3. EXPERIMENTS AND RESULTS

3.1. Data Preparation

To generate optical-flow data from RGB data, we used the implementation of the TV-L1 algorithm from the research of [11] in OpenCV library. For each RGB frame, we input its previous frame and itself to the algorithm, and the algorithm outputs one optical-flow frame that has the same size as the inputs and contains two channels for horizontal and vertical movements respectively.

For data augmentation, we used the same method that used in the research of Carreira, J. and Zisserman, A. [6]. Each video frame in the dataset has a size of 320 × 240 pixels. For the I3D model, we resized

Table 1. Behaviors description.

No.	Behavior name	Description
1	*drink*	The mouse drinks water from the water-feed nipple.
2	*eat*	The mouse eats food from the food-feed door.
3	*groom*	The mouse grooms its coat.
4	*hang*	The mouse hangs on the top of the cage.
5	*micro-movement*	The mouse slightly moves its head around.
6	*rear*	The mouse rears on the side of the cage.
7	*rest*	The mouse stays stable or sleeps. There is no movement at all.
8	*walk*	The mouse walks or runs inside the cage.

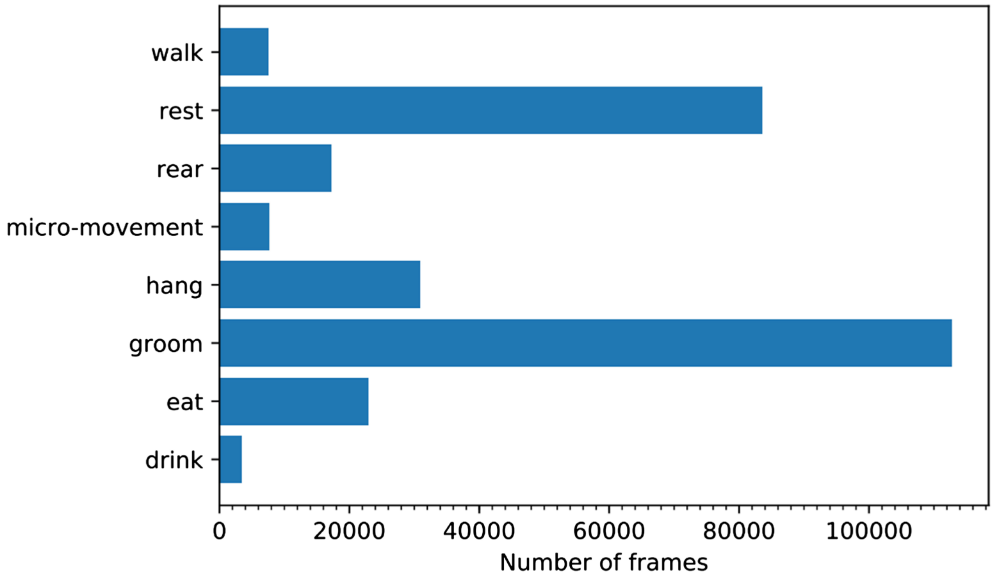


Figure 2. **Distribution of number of frames of each behavior in the "clipped database".**

them to size 255 × 255 pixels. Then we randomly cropped them to a size of 224 × 224 pixels and randomly horizontal flip them to create input frames. For R(2 + 1)D model, we resized the images to 128 × 128 pixels then randomly cropped them to a size of 112 × 112 pixels. This data augmentation method helps us to increase the accuracy of prediction of about 3%.

3.2. The Models

The I3D models are derived from Inception-V1 models [7]. To benefit from the 2D architecture, all filters and pooling kernels of 2D models were inflated to 3D by endowing them with an additional temporal dimension, *i.e.* $N \times N$ filters become $N \times N \times N$ filters. Then, the weights of 2D filters are repeated N times along the temporal dimension to bootstrap parameters from pre-trained 2D models to the 3D models. We showed the architecture of an inflated inception module used in I3D models in **Figure 3** and the detail of the architecture of the I3D model we used in this research in **Figure 5**.

The R(2 + 1)D models are derived from 2D versions [9] by replacing each 2D convolutional layer with two 3D convolutional layers, one for 2D image dimensions which have filters with size $1 \times N \times N$ and one for the time dimension which has filters with size $M \times 1 \times 1$. In some layers of the R(2 + 1)D models, to keep the total number of parameter to be the same as the 2D versions, the number of filter in these layers are calculated using the formula shown in **Figure 4**. The detail of the architecture of the R(2 + 1)D model we used in this research is shown in **Figure 5**.

For both models, we used 16 successive frames as an input (current frame, its 8 previous frames and its 7 next frames). To initialize parameters of the model, for the I3D models, we used weights from model-checkpoints that were pre-trained on ImageNet data [12]; and for the R(2 + 1)D models, we used weights from model-checkpoints that were pre-trained on Sport1M [13] and Kinetics data [14]. To fine-tune the models, we used momentum optimizer from the TensorFlow framework with momentum value equal to 0.9 and a learning rate start from 1e−3 and decay to 5e−5 after several thousands of iterations. We also

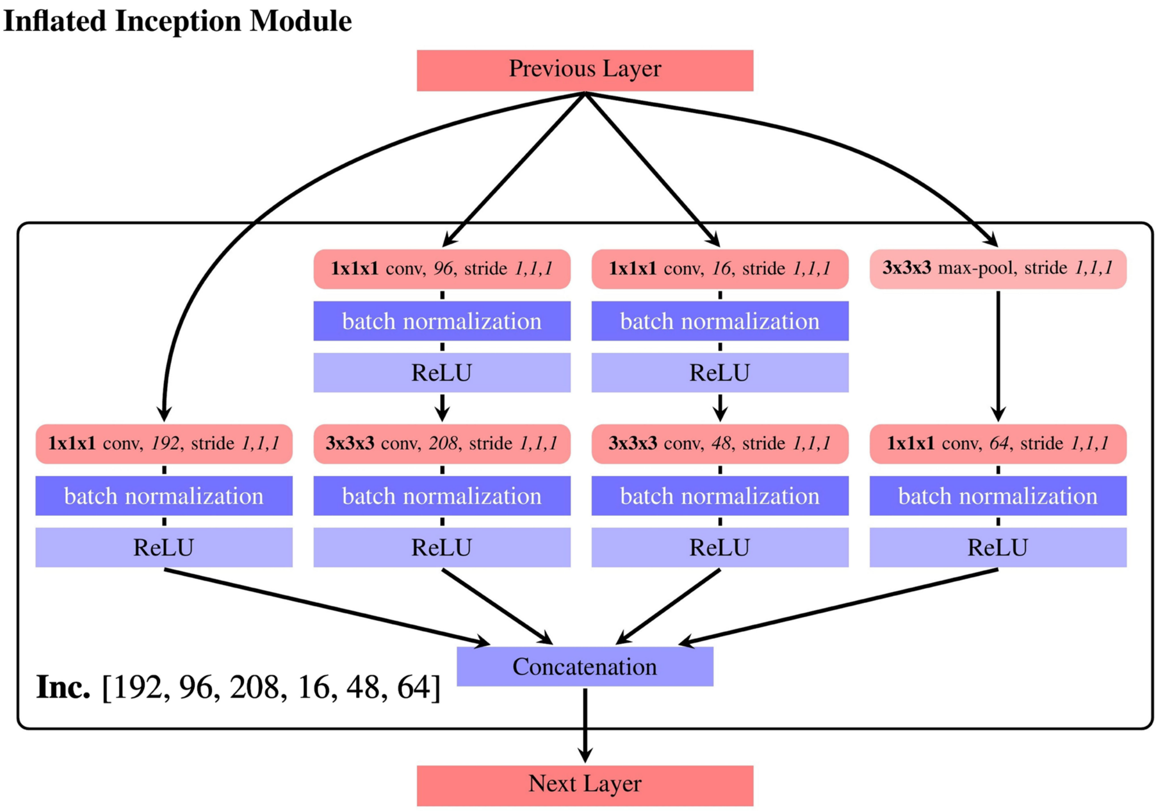
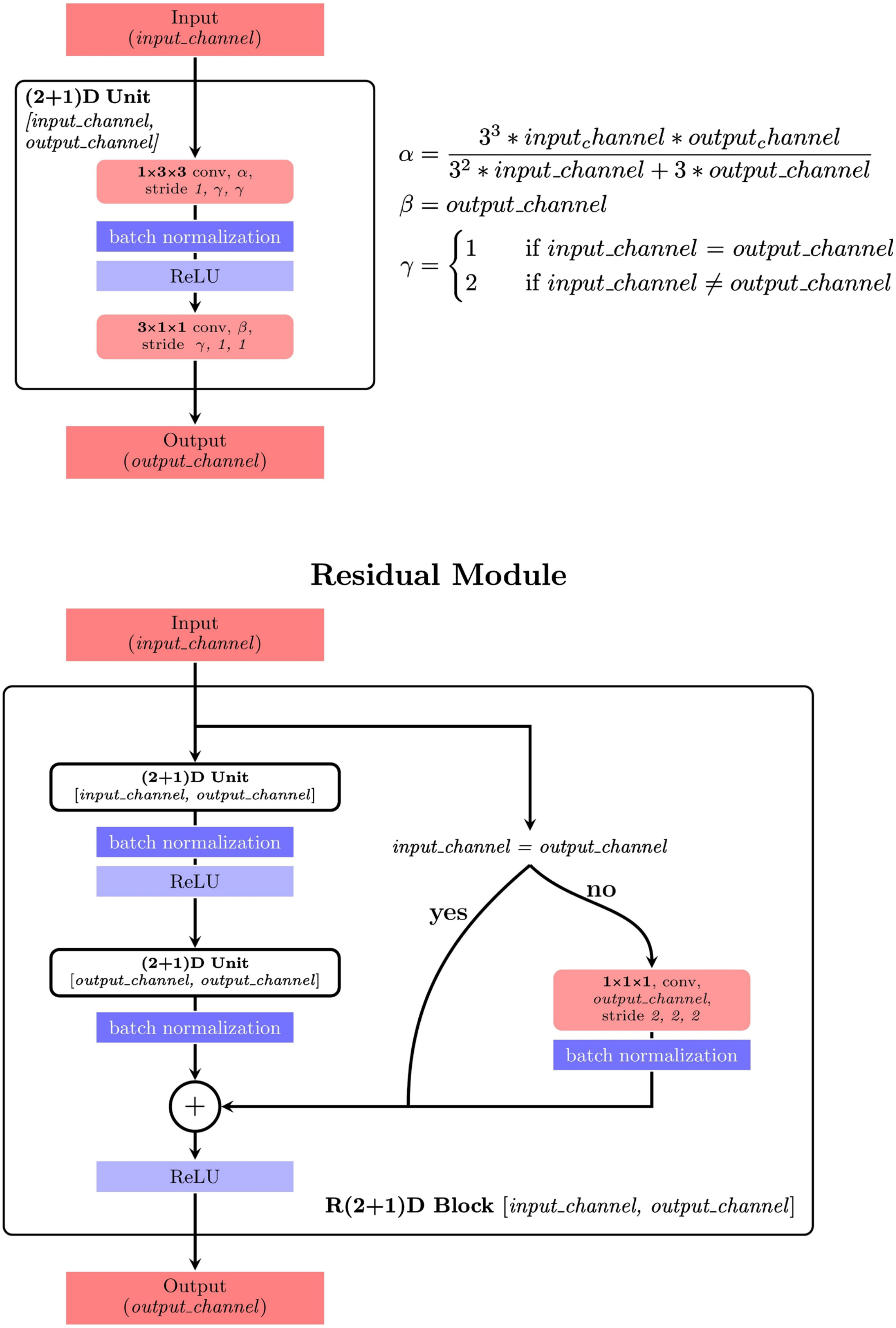

Inflated Inception Module
Previous Layer
1x1x1 conv, 96, stride 1,1,1
1x1x1 conv, 16, stride 1,1,1
3x3x3 max-pool, stride 1,1,1
batch normalization
ReLU
batch normalization
ReLU
1x1x1 conv, 192, stride 1,1,1
3x3x3 conv, 208, stride 1,1,1
3x3x3 conv, 48, stride 1,1,1
1x1x1 conv, 64, stride 1,1,1
batch normalization
batch normalization
batch normalization
batch normalization
ReLU
ReLU
ReLU
ReLU
Concatenation
Inc. [192, 96, 208, 16, 48, 64]
Next Layer

Figure 3. Architecture of an inflated inception module.



$$\alpha = \frac{3^3 * input_channel * output_channel}{3^2 * input\_channel + 3 * output\_channel}$$

$$\beta = output\_channel$$

$$\gamma = \begin{cases} 1 & \text{if } input\_channel = output\_channel \\ 2 & \text{if } input\_channel \neq output\_channel \end{cases}$$

Figure 4. Architecture of a (2 + 1)D residual module.

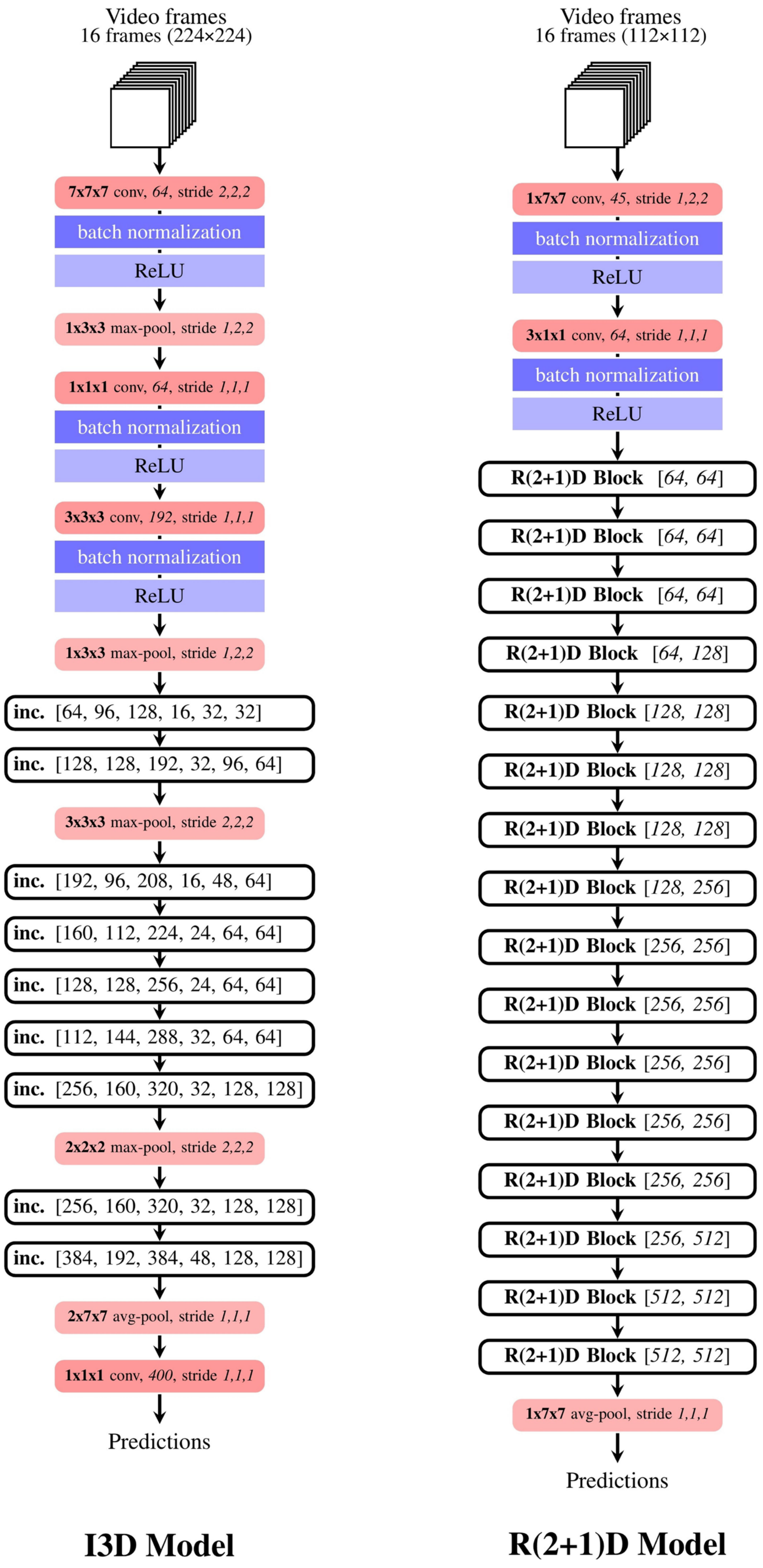

Video frames
16 frames (224×224)
7x7x7 conv, 64, stride 2,2,2
batch normalization
ReLU
1x3x3 max-pool, stride 1,2,2
1x1x1 conv, 64, stride 1,1,1
batch normalization
ReLU
3x3x3 conv, 192, stride 1,1,1
batch normalization
ReLU
1x3x3 max-pool, stride 1,2,2
inc. [64, 96, 128, 16, 32, 32]
inc. [128, 128, 192, 32, 96, 64]
3x3x3 max-pool, stride 2,2,2
inc. [192, 96, 208, 16, 48, 64]
inc. [160, 112, 224, 24, 64, 64]
inc. [128, 128, 256, 24, 64, 64]
inc. [112, 144, 288, 32, 64, 64]
inc. [256, 160, 320, 32, 128, 128]
2x2x2 max-pool, stride 2,2,2
inc. [256, 160, 320, 32, 128, 128]
inc. [384, 192, 384, 48, 128, 128]
2x7x7 avg-pool, stride 1,1,1
1x1x1 conv, 400, stride 1,1,1
Predictions
I3D Model
Video frames
16 frames (112×112)
1x7x7 conv, 45, stride 1,2,2
batch normalization
ReLU
3x1x1 conv, 64, stride 1,1,1
batch normalization
ReLU
R(2+1)D Block [64, 64]
R(2+1)D Block [64, 64]
R(2+1)D Block [64, 64]
R(2+1)D Block [64, 128]
R(2+1)D Block [128, 128]
R(2+1)D Block [128, 128]
R(2+1)D Block [128, 128]
R(2+1)D Block [128, 256]
R(2+1)D Block [256, 256]
R(2+1)D Block [256, 256]
R(2+1)D Block [256, 256]
R(2+1)D Block [256, 256]
R(2+1)D Block [256, 256]
R(2+1)D Block [256, 512]
R(2+1)D Block [512, 512]
R(2+1)D Block [512, 512]
1x7x7 avg-pool, stride 1,1,1
Predictions
R(2+1)D Model

Figure 5. Architecture of the I3D model and the R(2 + 1)D model.

used dropout in fully connected layers with keep-probability of 36% to reduce the effect of overfitting when fine-tuning the models.

As discussed in the paper of Carreira, J. and Zisserman, A. [6], although I3D models can learn motion features from RGB input videos, using optical-flow as inputs can introduce some recurrent sense to the models and significantly improved the performances. In this research, we also use the same fusion method to combine output predictions of I3D models and R(2 + 1)D models. The two-stream fusion method is illustrated in **Figure 6**. To investigate the effects of different two-stream fusion ratios in prediction performances, we tested various fusion ratios of the two models by setting different values for *rgb_weight* and *flow_weight* in the Two-stream fusion module. For example, if only using 30% of RGB data fine-tuned model's output and 70% of optical-flow data fine-tuned model's output then *rgb_weight* is set to 0.3 and *flow_weight* is set to 0.7.

Because frames of the dataset come from 12 different videos, we used leave-one-videos-out cross-validation to properly evaluate the performance of the models. For each video, we used all the frames extracted from it as testing data and all the frames extracted from the other videos as training data. We used the training data to fine-tune the models and used the fine-tuned models to predict labels for testing data. Then we count the total number of correct and incorrect prediction and calculate the accuracy.

3.3. Results

Figure 7 shows the results of using different fusion ratio of RGB and optical-flow data fine-tuned models on accuracies of prediction of each behavior. And **Figure 8** and **Figure 9** show confusion matrixes of correct and incorrect prediction ratio of behaviors in combinations of *rgb_weight* and *flow_weight*.

In **Figure 7**, we can see that for "*drink*" behaviors, combinations with more portion of RGB fine-tuned models have better performance than combinations with more portion of optical-flow fine-tuned models for both I3D models and R(2 + 1)D models. And the performance of R(2 + 1)D models

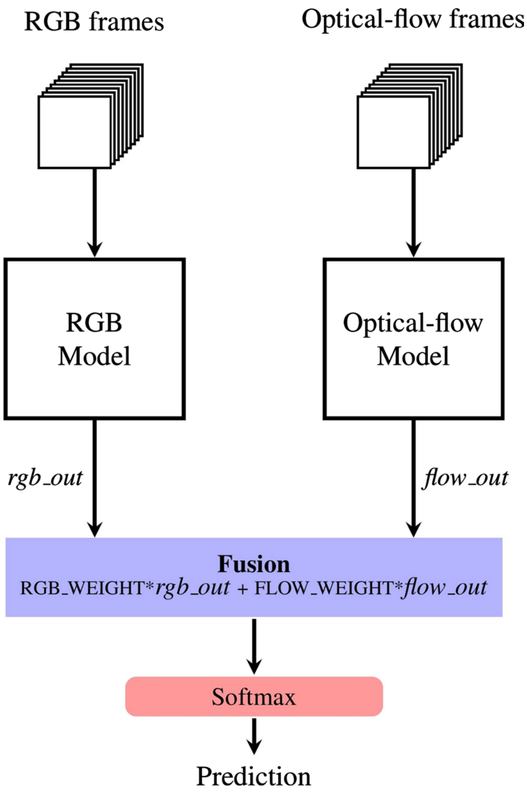


Figure 6. The two-stream fusion method.

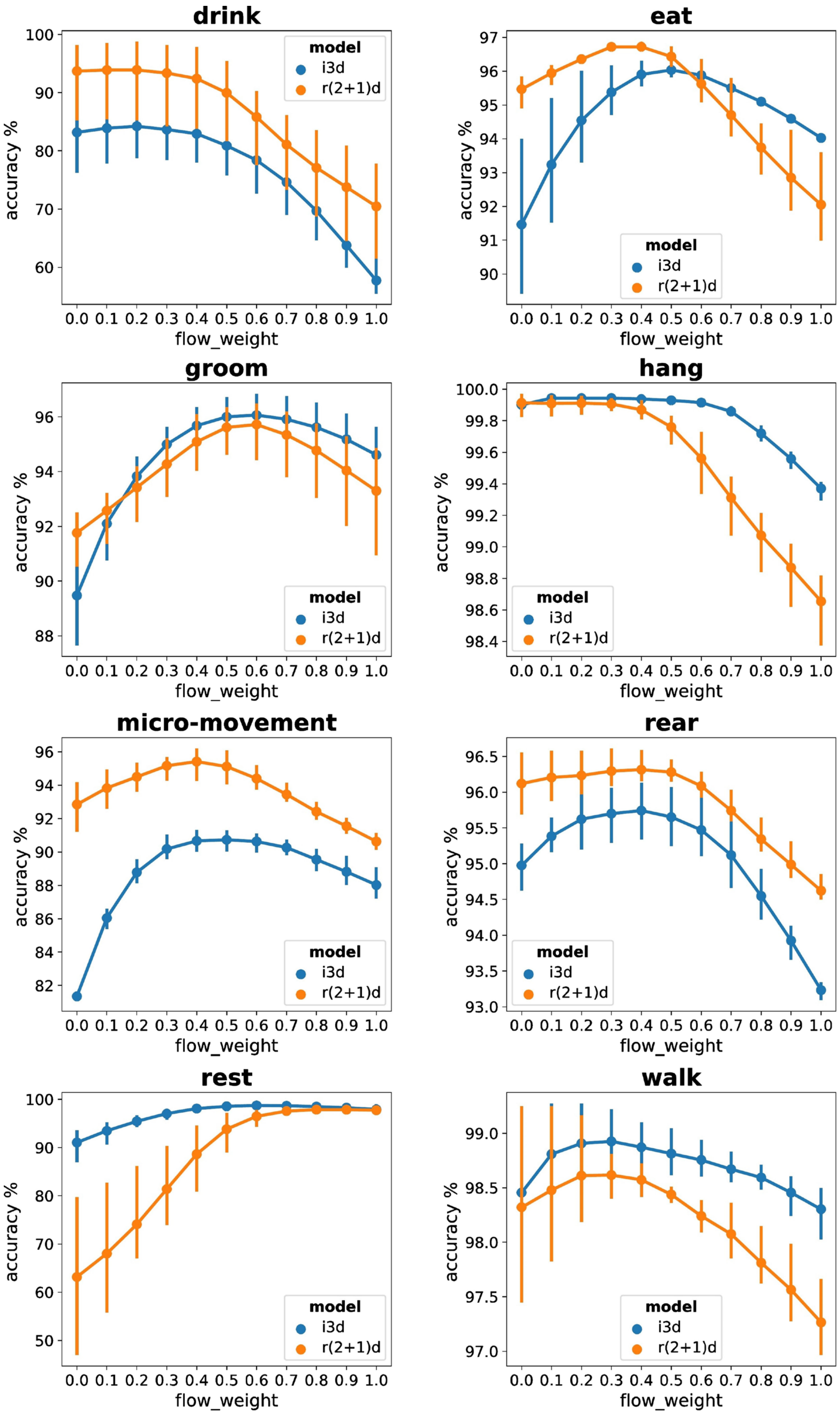
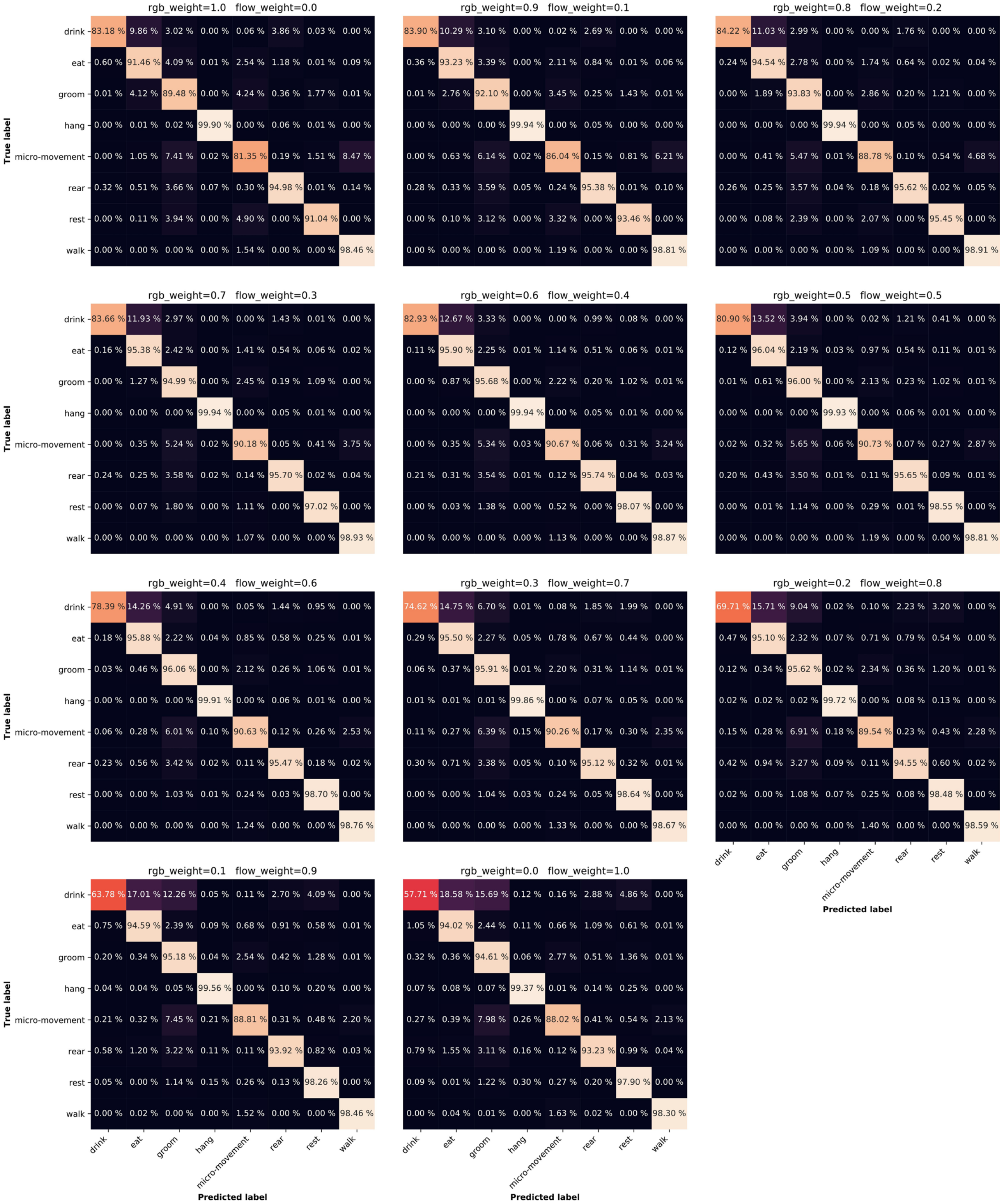


Figure 7. Accuracies of the prediction of each behavior with different two-stream fusion ratios.



rgb_weight=1.0 flow_weight=0.0

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	83.18 %	9.86 %	3.02 %	0.00 %	0.06 %	3.86 %	0.03 %	0.00 %
eat	0.60 %	91.46 %	4.09 %	0.01 %	2.54 %	1.18 %	0.01 %	0.09 %
groom	0.01 %	4.12 %	89.48 %	0.00 %	4.24 %	0.36 %	1.77 %	0.01 %
hang	0.00 %	0.01 %	0.02 %	99.90 %	0.00 %	0.06 %	0.01 %	0.00 %
micro-movement	0.00 %	1.05 %	7.41 %	0.02 %	81.35 %	0.19 %	1.51 %	8.47 %
rear	0.32 %	0.51 %	3.66 %	0.07 %	0.30 %	94.98 %	0.01 %	0.14 %
rest	0.00 %	0.11 %	3.94 %	0.00 %	4.90 %	0.00 %	91.04 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.54 %	0.00 %	0.00 %	98.46 %

rgb_weight=0.9 flow_weight=0.1

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	83.90 %	10.29 %	3.10 %	0.00 %	0.02 %	2.69 %	0.00 %	0.00 %
eat	0.36 %	93.23 %	3.39 %	0.00 %	2.11 %	0.84 %	0.01 %	0.06 %
groom	0.01 %	2.76 %	92.10 %	0.00 %	3.45 %	0.25 %	1.43 %	0.01 %
hang	0.00 %	0.00 %	0.00 %	99.94 %	0.00 %	0.05 %	0.00 %	0.00 %
micro-movement	0.00 %	0.63 %	6.14 %	0.02 %	86.04 %	0.15 %	0.81 %	6.21 %
rear	0.28 %	0.33 %	3.59 %	0.05 %	0.24 %	95.38 %	0.01 %	0.10 %
rest	0.00 %	0.10 %	3.12 %	0.00 %	3.32 %	0.00 %	93.46 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.19 %	0.00 %	0.00 %	98.81 %

rgb_weight=0.8 flow_weight=0.2

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	84.22 %	11.03 %	2.99 %	0.00 %	0.00 %	1.76 %	0.00 %	0.00 %
eat	0.24 %	94.54 %	2.78 %	0.00 %	1.74 %	0.64 %	0.02 %	0.04 %
groom	0.00 %	1.89 %	93.83 %	0.00 %	2.86 %	0.20 %	1.21 %	0.00 %
hang	0.00 %	0.00 %	0.00 %	99.94 %	0.00 %	0.05 %	0.00 %	0.00 %
micro-movement	0.00 %	0.41 %	5.47 %	0.01 %	88.78 %	0.10 %	0.54 %	4.68 %
rear	0.26 %	0.25 %	3.57 %	0.04 %	0.18 %	95.62 %	0.02 %	0.05 %
rest	0.00 %	0.08 %	2.39 %	0.00 %	2.07 %	0.00 %	95.45 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.09 %	0.00 %	0.00 %	98.91 %

rgb_weight=0.7 flow_weight=0.3

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	83.66 %	11.93 %	2.97 %	0.00 %	0.00 %	1.43 %	0.01 %	0.00 %
eat	0.16 %	95.38 %	2.42 %	0.00 %	1.41 %	0.54 %	0.06 %	0.02 %
groom	0.00 %	1.27 %	94.99 %	0.00 %	2.45 %	0.19 %	1.09 %	0.00 %
hang	0.00 %	0.00 %	0.00 %	99.94 %	0.00 %	0.05 %	0.01 %	0.00 %
micro-movement	0.00 %	0.35 %	5.24 %	0.02 %	90.18 %	0.05 %	0.41 %	3.75 %
rear	0.24 %	0.25 %	3.58 %	0.02 %	0.14 %	95.70 %	0.02 %	0.04 %
rest	0.00 %	0.07 %	1.80 %	0.00 %	1.11 %	0.00 %	97.02 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.07 %	0.00 %	0.00 %	98.93 %

rgb_weight=0.6 flow_weight=0.4

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	82.93 %	12.67 %	3.33 %	0.00 %	0.00 %	0.99 %	0.08 %	0.00 %
eat	0.11 %	95.90 %	2.25 %	0.01 %	1.14 %	0.51 %	0.06 %	0.01 %
groom	0.00 %	0.87 %	95.68 %	0.00 %	2.22 %	0.20 %	1.02 %	0.01 %
hang	0.00 %	0.00 %	0.00 %	99.94 %	0.00 %	0.05 %	0.01 %	0.00 %
micro-movement	0.00 %	0.35 %	5.34 %	0.03 %	90.67 %	0.06 %	0.31 %	3.24 %
rear	0.21 %	0.31 %	3.54 %	0.01 %	0.12 %	95.74 %	0.04 %	0.03 %
rest	0.00 %	0.03 %	1.38 %	0.00 %	0.52 %	0.00 %	98.07 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.13 %	0.00 %	0.00 %	98.87 %

rgb_weight=0.5 flow_weight=0.5

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	80.90 %	13.52 %	3.94 %	0.00 %	0.02 %	1.21 %	0.41 %	0.00 %
eat	0.12 %	96.04 %	2.19 %	0.03 %	0.97 %	0.54 %	0.11 %	0.01 %
groom	0.01 %	0.61 %	96.00 %	0.00 %	2.13 %	0.23 %	1.02 %	0.01 %
hang	0.00 %	0.00 %	0.00 %	99.93 %	0.00 %	0.06 %	0.01 %	0.00 %
micro-movement	0.02 %	0.32 %	5.65 %	0.06 %	90.73 %	0.07 %	0.27 %	2.87 %
rear	0.20 %	0.43 %	3.50 %	0.01 %	0.11 %	95.65 %	0.09 %	0.01 %
rest	0.00 %	0.01 %	1.14 %	0.00 %	0.29 %	0.01 %	98.55 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.19 %	0.00 %	0.00 %	98.81 %

rgb_weight=0.4 flow_weight=0.6

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	78.39 %	14.26 %	4.91 %	0.00 %	0.05 %	1.44 %	0.95 %	0.00 %
eat	0.18 %	95.88 %	2.22 %	0.04 %	0.85 %	0.58 %	0.25 %	0.01 %
groom	0.03 %	0.46 %	96.06 %	0.00 %	2.12 %	0.26 %	1.06 %	0.01 %
hang	0.00 %	0.00 %	0.00 %	99.91 %	0.00 %	0.06 %	0.01 %	0.00 %
micro-movement	0.06 %	0.28 %	6.01 %	0.10 %	90.63 %	0.12 %	0.26 %	2.53 %
rear	0.23 %	0.56 %	3.42 %	0.02 %	0.11 %	95.47 %	0.18 %	0.02 %
rest	0.00 %	0.00 %	1.03 %	0.01 %	0.24 %	0.03 %	98.70 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.24 %	0.00 %	0.00 %	98.76 %

rgb_weight=0.3 flow_weight=0.7

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	74.62 %	14.75 %	6.70 %	0.01 %	0.08 %	1.85 %	1.99 %	0.00 %
eat	0.29 %	95.50 %	2.27 %	0.05 %	0.78 %	0.67 %	0.44 %	0.00 %
groom	0.06 %	0.37 %	95.91 %	0.01 %	2.20 %	0.31 %	1.14 %	0.01 %
hang	0.01 %	0.01 %	0.01 %	99.86 %	0.00 %	0.07 %	0.05 %	0.00 %
micro-movement	0.11 %	0.27 %	6.39 %	0.15 %	90.26 %	0.17 %	0.30 %	2.35 %
rear	0.30 %	0.71 %	3.38 %	0.05 %	0.10 %	95.12 %	0.32 %	0.01 %
rest	0.00 %	0.00 %	1.04 %	0.03 %	0.24 %	0.05 %	98.64 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.33 %	0.00 %	0.00 %	98.67 %

rgb_weight=0.2 flow_weight=0.8

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	69.71 %	15.71 %	9.04 %	0.02 %	0.10 %	2.23 %	3.20 %	0.00 %
eat	0.47 %	95.10 %	2.32 %	0.07 %	0.71 %	0.79 %	0.54 %	0.00 %
groom	0.12 %	0.34 %	95.62 %	0.02 %	2.34 %	0.36 %	1.20 %	0.01 %
hang	0.02 %	0.02 %	0.02 %	99.72 %	0.00 %	0.08 %	0.13 %	0.00 %
micro-movement	0.15 %	0.28 %	6.91 %	0.18 %	89.54 %	0.23 %	0.43 %	2.28 %
rear	0.42 %	0.94 %	3.27 %	0.09 %	0.11 %	94.55 %	0.60 %	0.02 %
rest	0.02 %	0.00 %	1.08 %	0.07 %	0.25 %	0.08 %	98.48 %	0.00 %
walk	0.00 %	0.00 %	0.00 %	0.00 %	1.40 %	0.00 %	0.00 %	98.59 %

rgb_weight=0.1 flow_weight=0.9

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	63.78 %	17.01 %	12.26 %	0.05 %	0.11 %	2.70 %	4.09 %	0.00 %
eat	0.75 %	94.59 %	2.39 %	0.09 %	0.68 %	0.91 %	0.58 %	0.01 %
groom	0.20 %	0.34 %	95.18 %	0.04 %	2.54 %	0.42 %	1.28 %	0.01 %
hang	0.04 %	0.04 %	0.05 %	99.56 %	0.00 %	0.10 %	0.20 %	0.00 %
micro-movement	0.21 %	0.32 %	7.45 %	0.21 %	88.81 %	0.31 %	0.48 %	2.20 %
rear	0.58 %	1.20 %	3.22 %	0.11 %	0.11 %	93.92 %	0.82 %	0.03 %
rest	0.05 %	0.00 %	1.14 %	0.15 %	0.26 %	0.13 %	98.26 %	0.00 %
walk	0.00 %	0.02 %	0.00 %	0.00 %	1.52 %	0.00 %	0.00 %	98.46 %

rgb_weight=0.0 flow_weight=1.0

	drink	eat	groom	hang	micro-movement	rear	rest	walk
drink	57.71 %	18.58 %	15.69 %	0.12 %	0.16 %	2.88 %	4.86 %	0.00 %
eat	1.05 %	94.02 %	2.44 %	0.11 %	0.66 %	1.09 %	0.61 %	0.01 %
groom	0.32 %	0.36 %	94.61 %	0.06 %	2.77 %	0.51 %	1.36 %	0.01 %
hang	0.07 %	0.08 %	0.07 %	99.37 %	0.01 %	0.14 %	0.25 %	0.00 %
micro-movement	0.27 %	0.39 %	7.98 %	0.26 %	88.02 %	0.41 %	0.54 %	2.13 %
rear	0.79 %	1.55 %	3.11 %	0.16 %	0.12 %	93.23 %	0.99 %	0.04 %
rest	0.09 %	0.01 %	1.22 %	0.30 %	0.27 %	0.20 %	97.90 %	0.00 %
walk	0.00 %	0.04 %	0.01 %	0.00 %	1.63 %	0.02 %	0.00 %	98.30 %

Figure 8. Confusion matrices of predictions of I3D models with different two-stream fusion ratios.

are better than the performance of the I3D models in this behaviors. In Figure 8 and Figure 9, from confusion matrixes of both I3D model and R(2 + 1)D model, we can see that almost false predicted samples of "*drink*" behaviors are misclassified as "*eat*" behaviors. We can explain that as the water-feed nipple and

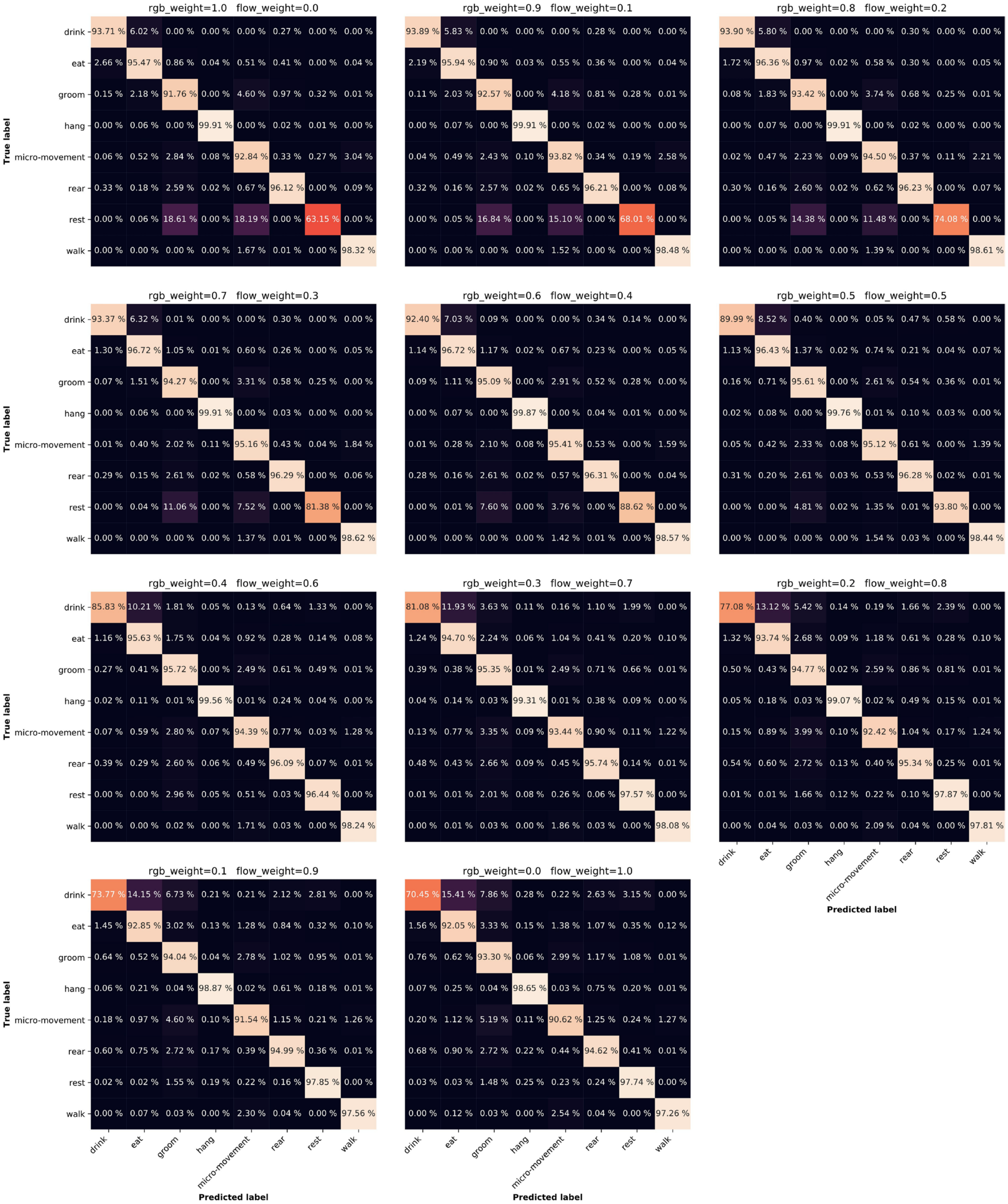


Figure 9. Confusion matrices of predictions of R(2 + 1)D models with different two-stream fusion ratios.

food-feed door are quite close to each other; Sometimes the two behaviors look very similar and the dataset is also imbalanced with the ratio of the number of "*drink*" frames to the number of "*eat*" frames is about 1:6.85. Therefore, the models tend to predict "*drink*" behaviors as "*eat*" behaviors in many cases. It also explains why models fine-tuned using RGB data are more precise in distinguishing the two behaviors than models fine-tuned using optical-flow data. RGB data fine-tuned models can utilize the information of the mouse's mouth contact with water-feed nipple or food-feed door. However, this information is lost in optical-flow data because there is no motion of water-feed nipple or food-feed door in the scenes.

For "*eat*", "*groom*", "*micro-movement*", "*rear*", and "*walk*" behaviors, we can see that using a right combination of RGB data fine-tuned models and optical-flow data fine-tuned models give a better performance than using these models only. The R(2 + 1)D model outperforms I3D model in classifying "*micro-movement*" and "*rear*" behaviors but the I3D model is better in classifying "*walk*" behaviors.

The two models work very well on classifying "*hang*" behaviors and their performances just slightly reduce when we use a high portion of optical-flow data fine-tuned models because of the lack of the mouse surrounding environment in the optical-flow data.

And for the "*rest*" behaviors, it is easy to understand why using a high portion of optical-flow data fine-tuned models give better performance as "*rest*" behaviors are different from other behaviors that they have no movement in the scenes.

Overall, the two deep learning models we investigated in this research outperform the previous research model in the Mouse behavior dataset as shown in Table 2. The accuracies of the two models with different fusion ratio are shown in Figure 10. Both models have best performances at fusion ratio of

Table 2. Comparison of performance of models.

No	Model	Accuracy (%)
1	MF + SVMHMM [1]	93.0
2	FV + NN [4]	95.9
3	I3D	**96.9**
4	R(2 + 1)D	96.3

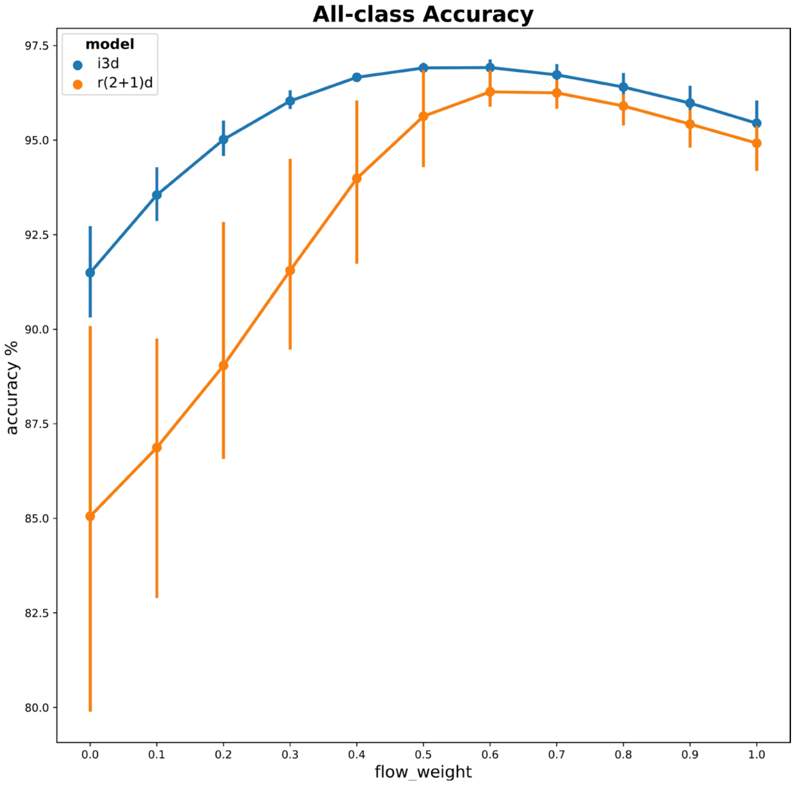


Figure 10. Accuracies of I3D models and R(2 + 1)D models with different two-stream fusion ratios.

40% RGB data fine-tuned models and 60% optical-flow data fine-tuned models. The I3D model achieves 96.9% of accuracy and the R(2 + 1)D achieves 96.3% of accuracy.

4. CONCLUSIONS

We have investigated two current state-of-the-art deep learning models for human action recognition in a mouse behavior recognition task. Both models outperformed the models from previous researches. It proves that our approach of utilizing deep learning models that pre-trained on human action datasets and fine-tuning them for animal behavior recognition tasks is efficient despite the scarcity of training data. We also showed the effect of two-stream fusion ratios on the predictions.

The fine-tuned models can precisely recognize most of behaviors they learned from the mouse behavior dataset. But there are some difficulties in classifying behaviors that are ambiguous or similar to other behaviors. Our proposal to solve the problem is to collect more data on difficult-to-classify behaviors. And we can redesign experimental environment such as changing the camera position or the cage configuration in order to minimize the ambiguity between behaviors.

For further research, we will collect behavior data of other animals. Then we will use them to fine-tune the fine-tuned models we achieved from this research to experiment if we can really efficiently utilize deep learning models for animal behavior recognition tasks without any requirements of extensive data for training these models.

ACKNOWLEDGEMENTS

In this research, the super-computing resource was provided by Human Genome Center, the Institute of Medical Science, The University of Tokyo. Additional computation time was provided by the super computer system in Research Organization of Information and Systems (ROIS), National Institute of Genetics (NIG). This work was supported by JSPS KAKENHI Grant Number JP18K11525.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest regarding the publication of this paper.

REFERENCES

1. Jhuang, H., Garrote, E., Yu, X., Khilnani, V., Poggio, T., Steele, A.D. and Sere, T. (2010) Automated Home-Cage Behavioural Phenotyping of Mice. *Nature communications*, **1**, Article Number: 68. https://doi.org/10.1038/ncomms1064

2. Jhuang, H., Serre, T., Wolf, L. and Poggio, T. (2007) A Biologically Inspired System for Action Recognition. 2007 *IEEE 11th International Conference of Computer Vision*, Rio de Janeiro, 14-21 October 2007, 716-725. https://doi.org/10.1109/ICCV.2007.4408988

3. Altun, Y., Tsochantaridis, I. and Hofmann, T. (2003) Hidden Markov Support Vector Machines. *International Conference on Machine Learning*, Washington DC, 21-24 August 2003, 3-10.

4. Jiang, Z., Crokes, D., Green, B.D., Zhang, S. and Zhou, H. (2017) Behaviour Recognition in Mouse Videos Using Contextual Features Encoded by Spatial-Temporal Stacked Fisher Vectors. *International Conference on Pattern Recognition Applications and Methods*, 259-269. https://doi.org/10.5220/0006244602590269

5. Dollar, P., Rabaud, V., Cottrell, G. and Belongie, S. (2005) Behavior Recognition via Sparse Spatio-Temporal Feature. *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Beijing, 15-16 October 2005, 65-72. https://doi.org/10.1109/VSPETS.2005.1570899

6. Carreira, J. and Zisserman, A. (2018) Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. 2017 *IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, 21-26 July 2017, 4724-4733.

https://doi.org/10.1109/CVPR.2017.502

7. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015) Going Deeper with Convolutions. 2015 *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, 7-12 June 2015, 1-9. https://doi.org/10.1109/CVPR.2015.7298594

8. Tran, D., Wang, H., Torrensani, L., Ray, J., LeCun, Y. and Paluri, M. (2018) A Closer Look at Spatiotemporal Convolutions for Action Recognition. *Computer Vision and Pattern Recognition*. https://arxiv.org/abs/1711.11248

9. He, K., Zhang, X., Ren, S. and Sun, J. (2015) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 27-30 June 2016, 770-778. https://doi.org/10.1109/CVPR.2016.90

10. Torrey, L. and Shavlik, J. (2009) Transfer Learning. In: Soria, E., Martin, J., Magdalena, R., Martinez, M. and Serrano, A., Eds., *Handbook of Research on Machine Learning Applications*, IGI Global, 242-264.

11. Zach, C., Pock, T. and Bischof, H. (2007) A Duality Based Approach for Realtime TV-L^1 Optical Flow. *Proceeding of 29th DAGM Symposium of Pattern Recognition*, **4713**, 214-223. https://doi.org/10.1007/978-3-540-74936-3_22

12. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C. and Fei, L.F. (2015) ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, **115**, 211-252. https://doi.org/10.1007/s11263-015-0816-y

13. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R. and Fei, L.F. (2014) Large-Scale Video Classification with Convolutional Neural Networks. 2014 *IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, 23-28 June 2014, 1725-1732. https://doi.org/10.1109/CVPR.2014.223

14. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M. and Zisserman, A. (2017) The Kinetics Human Action Video Dataset. *Computer Vision and Pattern Recognition*. https://arxiv.org/abs/1705.06950