Heliyon 9 (2023) e16763

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

Vision-based activity recognition in children with autism-related behaviors

Pengbo Wei [a], David Ahmedt-Aristizabal [a,b,*,1], Harshala Gammulle [a,b], Simon Denman [b], Mohammad Ali Armin [a]

[a] *Imaging and Computer Vision Group, CSIRO Data61, Canberra, Australia*
[b] *SAIVT, Queensland University of Technology, Brisbane, Australia*

ARTICLE INFO

Dataset link: https://github.com/Samwei1/autism-related-behavior

ABSTRACT

Advances in machine learning and contactless sensors have enabled the understanding complex human behaviors in a healthcare setting. In particular, several deep learning systems have been introduced to enable comprehensive analysis of neuro-developmental conditions such as Autism Spectrum Disorder (ASD). This condition affects children from their early developmental stages onwards, and diagnosis relies entirely on observing the child's behavior and detecting behavioral cues. However, the diagnosis process is time-consuming as it requires long-term behavior observation, and the scarce availability of specialists. We demonstrate the effect of a region-based computer vision system to help clinicians and parents analyze a child's behavior. For this purpose, we adopt and enhance a dataset for analyzing autism-related actions using videos of children captured in uncontrolled environments (*e.g.* videos collected with consumer-grade cameras, in varied environments). The data is pre-processed by detecting the target child in the video to reduce the impact of background noise. Motivated by the effectiveness of temporal convolutional models, we propose both light-weight and conventional models capable of extracting action features from video frames and classifying autism-related behaviors by analyzing the relationships between frames in a video. By extensively evaluating feature extraction and learning strategies, we demonstrate that the highest performance is attained through the use of an Inflated 3D Convnet and Multi-Stage Temporal Convolutional Network. Our model achieved a Weighted F1-score of 0.83 for the classification of the three autism-related actions. We also propose a light-weight solution by employing the ESNet backbone with the same action recognition model, achieving a competitive 0.71 Weighted F1-score, and enabling potential deployment on embedded systems. Experimental results demonstrate the ability of our proposed models to recognize autism-related actions from videos captured in an uncontrolled environment, and thus can assist clinicians in analyzing ASD.

* Corresponding author.
E-mail addresses: sam.wei@data61.csiro.au (P. Wei), david.ahmedtaristizabal@data61.csiro.au (D. Ahmedt-Aristizabal), pranali.gammule@qut.edu.au (H. Gammulle), s.denman@qut.edu.au (S. Denman), ali.armin@data61.csiro.au (M.A. Armin).
[1] Same contribution as the first author.

https://doi.org/10.1016/j.heliyon.2023.e16763
Received 12 May 2023; Received in revised form 25 May 2023; Accepted 26 May 2023

Available online 1 June 2023

1. Introduction

A computational understanding of human behaviors has significant potential for applications across multiple domains, including healthcare [1]. However, modeling and automatically analyzing human behaviors is extremely challenging, as behaviors are contextual and often social, *i.e.* in relation to other people. Thus, an understanding of interaction dynamics may be needed to understand an individual's behavior. An important application area of AI-enabled computational behavior analysis is in characterizing the behavior and developmental changes in children who are diagnosed with Autism Spectrum Disorder (ASD).

ASD is a neurodevelopmental disorder characterized by a set of social communication deficits, self-harm, or persistent repetition of actions [2]. Moreover, this disorder often manifests in children during their early developmental stages and can have severe negative impacts on the quality of their life over a long time period [3]. At present in Australia, in excess of 1 in 150 children are diagnosed with ASD [4]. There can also be a considerable delay in ASD diagnosis from the point at which parents sought professional help [5]. There are two reasons for the long wait times for ASD diagnosis; i) the low availability of specialists, ii) there are no reliable biomarkers for ASD, and its diagnosis requires a long-term observation of stereotypical behaviors such as headbanging, arm-flapping or spinning. It is worth noting that in this study, we analyze stereotypical behaviors that may have ASD related biomarkers such as those determined through functional magnetic resonance imaging (*f*MRI). Facial expressions [6], eye gaze, and motor control/movement patterns are outside the scope of this study due to privacy-preserving concerns. Interested readers are referred to [7] for more information.

Computational methods have been effectively employed to assist with the identification of ASD (*i.e.* to differentiate the body dynamics of children with ASD from their typically developing peers) and for the classification of different ASD behavioral biomarkers through video analysis, being the latter the focus of this work. Such a classification can be performed through dynamic [8–10] or static analysis [11]).

A pioneering computer vision based method for analyzing self-stimulatory behaviors for ASD from video was proposed by Rajagopalan et al. [12]. The authors extracted image features and developed a system based on a standard Bag of Words (BOW). Subsequently, Rajagopalan and Goecke [13] modeled self-stimulatory behaviors by selecting poselet bounding boxes and a high-level global descriptor that captured dominant motion patterns in video. Recently, Kojovic et al. [9] analyzed ASD behaviors using a spatial-temporal model that consists of a VGG16 model [14] and a long short-term memory (LSTMs) [15] neural network using 2D human pose data extracted from sequential frames. However, in a typical video dataset, samples often include additional irrelevant people. The presence of these people can hamper the action recognition model during training. Thus, Washington et al. [16], extracted the target head pose from videos with a head banging detector, and then analyze the head banging action using a similar hybrid convolutional neural network (CNN) and LSTM architecture. Similarly, a 3D-CNN based on an Inflated 3D ConvNet (I3D) [17] along with target person detection and tracking techniques are adopted by Ali et al. [10] for action recognition tasks in videos of children with ASD. Negin et al. [18] investigated two approaches and compared a bag-of-visual-words approach with recurrent neural networks (RNNs) and CNNs, and showed that deep learning architectures offer competitive performance for the classification of four actions including arm flapping, headbanging, hand actions and spinning. Furthermore, in order to address the challenges of limited data, Pandey et al. [19] introduced a guided weak supervision technique to match the target data with a class in the source data. This was followed by training a classifier on the augmented data to classify autism-related behaviors.

While the dynamic analysis of videos (where the temporal nature of the data is expressly modeled) are more common, some researchers have developed static approaches for ASD behavioral analysis. Liang et al. [11] employed an unsupervised model, as Temporal Coherency Deep Networks (TCND), to classify actions. The TCND model involves four ALexNet models [20], all having the same parameters, to extract features from frames. After the feature extraction, a linear support vector machines (SVMs) is used to classify actions based on the extracted features. This is an example of a static approach which disregards temporal features and classifies an individual video frame. Results at the video level are reported based on the average obtained by applying the approach to each frame.

In general, Temporal Convolutional Networks (TCN) have shown better performance in comparison to LSTMs for various tasks including action recognition [21]. Wang et al. [22] demonstrated that the combination of a 3D-CNN and temporal models can increase Human Action Recognition performance. However, the related works to analyze ASD behaviors focused on adopting very highly parameterized models, thus limiting the ability to deploy such a method on an embedded device. To overcome this limitation, we deployed light-weight models for the classification of ASD behaviors in complex environments, that can be applied to videos collected with consumer grade cameras.

Apart from aforementioned framework challenges, an appropriate dataset is another important component needed to address ASD monitoring. Rajagopalan et al. [12] introduced a new **S**elf-**S**timulatory **B**ehavior **D**ataset (SSBD) for ASD. This dataset was collected from YouTube videos recorded in uncontrolled environments, and included three stereotypical behaviors including arm flapping, headbanging and spinning. Compared with traditional action recognition datasets which consists of tens of thousands of action videos (*e.g.* Kinetics [23]), this dataset which contains only 75 videos can be considered a shallow dataset. Given the limited size of the dataset, Negin et al. [18] introduced an Expanded Stereotype Behavior Dataset (ESBD) that includes an additional behavior (hand action), collected from YouTube. The ESBD dataset consists of 141 video and does not share any videos with the SSBD dataset. However, it is important to note that this dataset has not been made publicly available at this time.

In addition to SSBD, we present a stereotypical behavior dataset that incorporates potential autism-related videos collected from online media platforms. With this extended action recognition dataset, we have developed spatio-temporal frameworks that use computer vision techniques to classify stereotypical behaviors under light-weight and standard operating conditions, and that can be directly adapted for any related datasets containing atypical behaviors. Our proposed method involves two main streams: a feature

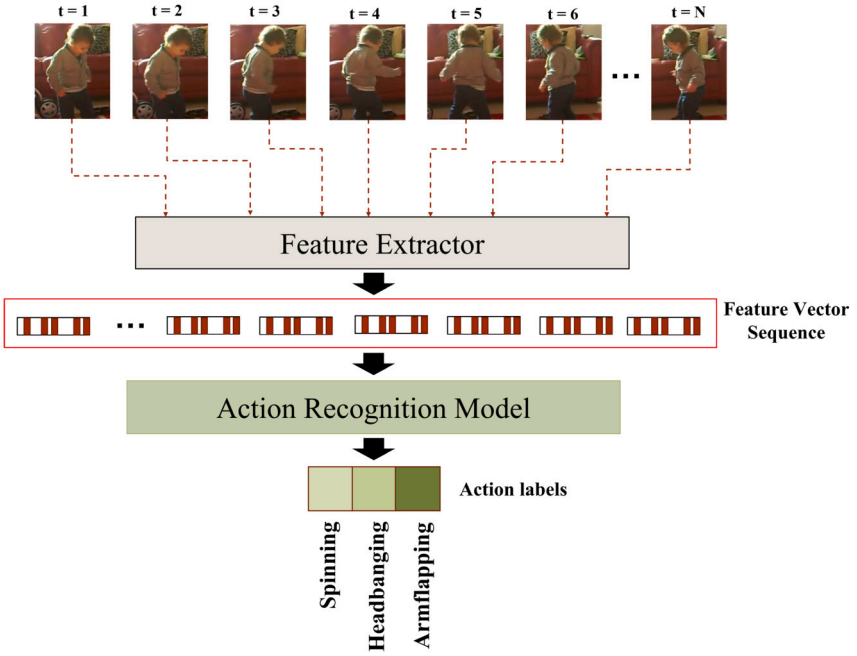

Fig. 1. Overview of the activity recognition pipeline. Given a sequence of RGB video frames, the system generates a sequence of feature vectors via the feature extractor. Next, the action recognition model recognizes the activity by using the generated sequence of visual features.

extractor that utilizes CNNs to extract embedding features from video frames depicting autism, and a temporal model that detects and classifies each action based on the extracted embeddings.

Our main contributions are summarized as follows:

1. We curate an existing dataset of stereotypical children's autism behavior videos recorded in an uncontrolled environment by incorporating 12 new subjects, and removing 11 original noisy subjects. The new dataset contains 61 videos and is divided into 168 short video clips. Information of each video (URL and type) is available[2] to facilitate greater reproducibility.
2. To leverage the ASD stereotypical behavior prior knowledge and improve the robustness, we exploit region-based action recognition frameworks that utilize conventional and light-weight architectures combined with multi-stage temporal convolutional networks to recognize characteristic autism behaviors, achieving superior performance in comparison to traditional models. It is worth to be noted that we assess a compressed feature representation to examine if it enables real-time throughput while maintaining acceptable accuracy.

2. Methods

In this section, we describe the proposed region-based deep learning approach that recognizes autism-related behaviors in children. Fig. 1 illustrates the overall architecture of our framework, which has two crucial steps: **i**) a feature extractor that extracts frame-wise action-related semantic features; and **ii**) an action recognition model which recognizes autism-related behaviors by modeling the temporal evolution of the extracted frame-wise features. As it is challenging to directly compare our work with previous ASD action recognition work due to the unavailability of code and datasets, we favorably compare the performance of seven common feature extractors and four learning strategies that have been employed for general action recognition and ASD tasks. A more detailed explanation of the feature extraction and action recognition models that were used in our experiments is provided in the following subsections. We further investigate different combinations of these feature extraction and action recognition models to recognize autism-related behaviors efficiently.

2.1. Feature extraction backbones

Feature extraction is a crucial step in the proposed action recognition pipeline, and is responsible for extracting embeddings from raw frames. As well as helping to capture the most salient features for the task at hand, this process reduces image dimensionality (*e.g.* an image of size 224×224 may be represented only by a vector of size 1024), which can further enhances the action learning process by removing redundant data.

[2] https://github.com/Samwei1/autism-related-behavior/blob/main/url_list.pdf.

Table 1
Pros and cons of selected feature extraction backbones.

Backbone	Pros	Cons
EfficientNet	- Computationally efficient compared to ResNet, I3D, etc. - Robust features due to the compound scaling mechanism.	- Not suitable for mobile applications.
MobileNet	- Computationally efficient. - Does not require an impressive amount of data to train. - Suitable for mobile and embedded vision applications. - Low computational resources are required.	- Limited model capacity to learn large-scale data. - Produce less accurate results compared to EfficientNet, ResNet and I3D
ShuffleNet	- Computationally efficient than MobileNet. - Does not require an impressive amount of data to train. - Suitable for mobile and embedded vision applications. - Low computational resources are required. - More accurate than MobileNet.	- Limited model capacity to learn largescale complex data.
ESNet	- Computationally efficient than MobileNet and ShuffleNet. - Real-time performance. - Suitable for mobile and embedded vision applications. - Does not require an impressive amount of data to train. - Low computational resources are required.	- Limited model capacity to learn largescale complex data.
ResNet	- Produce more accurate results than the shallow networks - Suitable for learning large-scale complex data. - The residual connections overcome the vanishing gradient problem.	- Requires more memory and computational resources, which makes it less suitable for mobile applications.
I3D	- More suitable for video-based applications to capture spatio-temporal features. - Suitable for learning large-scale complex data. - Produce more accurate results than shallow networks such as MobileNet etc.	- Requires more memory and computational resources, which makes it less suitable for mobile applications.

Traditional feature extraction methods (such as spatio-temporal interest points (STIP) used in [12]) require feature engineering and rely on substantial domain knowledge of human experts to obtain best results. In contrast, deep learning models learn features automatically and have been used to achieve better performance for feature extraction and classification tasks [24–27]. Once the inputs and the corresponding labels are given, deep networks are able to learn a mapping that captures the relationship between the input and the output data automatically. However, learning this mapping requires a great deal of data and time to train a deep network from scratch. When the requisite amount of data is not available, a common practice is to adopt transfer learning, where we reuse a model that has previously been trained on a large dataset and fine-tune the network (or a portion of the network) with our target dataset. Considering these approaches, in our work, we investigate using both pre-trained and fine-tuned feature extractors to determine the best feature extractor for the behavior recognition task.

We select feature extractors appropriate for two different operating conditions: i) *a light-weight environment* condition suitable for an embedded device; and ii) *a standard operating* condition (*i.e.* conventional models) suitable for a powerful workstation or server. We investigated the following feature extractors: EfficientNets [24] MobileNet [26], ShuffleNet [28], Enhanced ShuffleNet (ESNet) [29], ResNet [30], and Inflated 3D ConvNet (I3D) [17]. The models EfficientNet-B0, MobileNetV3, ShuffleNetV2 and ESNet are identified as light-weight models (number of model parameters < 5.5 M), while EfficientNet-B3, ResNet18 and I3D are considered as the standard operating condition (number of model parameters < 15 M). Number of parameters are described in millions in this manuscript. It should be noted that the *standard operating condition* models themselves are still substantially less computationally demanding compared to the two-stream models utilized in [10], or more complex networks in the literature such as ResNet101_vd (44 M), HRNet_W48 (78 M), and SwinTransformer (99 M). In [10], the authors use I3D models for both RGB and Optical flow inputs, whereas in our work, only a single I3D model is applied to the RGB frames.

When choosing feature extraction models, we focused on selecting light-weight models with a low number of trainable parameters. The main reason for this is to support a real-world application by improving the efficiency of the feature extraction step and helping to reduce data requirements for fine-tuning. In addition, deeper networks can struggle during training due to problems such as vanishing gradients caused by the increased network depth. The following subsections discuss the technical details of the feature extraction networks (backbones) and Table 1 further summarizes these details.

2.1.1. EfficientNets

EfficientNets [24] are composed of a deep CNN architecture with fewer parameters compared to previous models such as VGG and AlexNet. EfficientNets are scaled by an effective compound scaling mechanism, where the network is uniformly scaled in all dimensions including depth, width, and resolution, from a baseline EfficientNet architecture that is generated by a neural architecture

search. Among their models, EfficientNet-B0 is the simplest and most efficient model. Despite being light-weight, EfficientNet-B0 has achieved the state-of-the-art performance on image classification tasks with fewer parameters on average than other CNN models, such as AlexNet [20], Inception-v2 [31], and ResNet-50 [32]. We also consider EfficientNet-B3, which achieves improved performance over previous architectures but with fewer parameters (81.1% accuracy on ImageNet [33] with 12M parameters). Critically, the transfer learning performance of EfficientNet has also been shown to be better than other architectures [24].

2.1.2. MobileNets

MobileNets are light-weight CNN models based on a streamlined architecture that uses depth-wise separable convolutions to build light weight yet deep neural networks. They are widely used in mobile and embedded vision applications and achieve relatively high performance for object detection, fine-grained classification, face attribute classification, and large scale geo-localization tasks [26]. In this research, we adopt MobileNetV3 [34] which is more accurate for image classification tasks than previous versions of MobileNet, while also being more computationally efficient.

2.1.3. ShuffleNet

ShuffleNet [35] is a light-weight model that is widely adopted in portable and low computation applications. To solve issues that exist in other light-weight networks (a larger number of feature channels causing high computation complexity), Zhang et al. [35] proposed two techniques, pointwise group convolutions and bottleneck-like structures. However, the two techniques were found to be costly for light-weight networks. To further reduce the computation complexity of ShuffleNet, Ma et al. [28] introduced the channel split operator to help maintain a large number and equally wide set of channels while avoiding dense convolution and having too many groups of convolution operations in networks, and thus proposed the ShuffleNetV2.

2.1.4. ESNet

The Enhanced ShuffleNet (ESNet) [29] further improves the network structure of ShuffleNetV2 [28] for real-time object detectors, with a better balance between accuracy and latency on mobile devices. ESNet adapts some methods used by the lightweight CPU network PP-LCNet [36], with additional modifications. The Squeeze-and-excitation module (SE) [37] is a channel attention module, which is also used in MobileNetv3, and is included in all blocks of ESNet to improve the feature extraction by weighting the network channels. Furthermore, the author of GhostNet [38] proposes a novel Ghost module that can generate more feature maps with fewer parameters to improve the network's learning ability. The Ghost module is also added in the blocks with stride set to 1 to further enhance the performance of the ESNet.

2.1.5. ResNet

Recent studies have shown the high performance of ResNet [32], which leverages the residual block structure, for image recognition and classification tasks. It uses residual (or skip) connections to allow information to more readily propagate through the network. To improve the accuracy and robustness of this popular network without increasing the amount of computation, the authors in [30] proposed an improved method ResNet-vd with a collection of refinements, such as modifying the downsampling operation, which can generate better feature maps. ResNet models have many variants with different number of layers and different residual block structures. In this study, we investigated ResNEt18-vd which consists of 18 residual blocks and 11.7M parameters.

2.1.6. Inflated 3D convnet (I3D)

Inception-VI was used as the backbone network for the I3D model [17] by expanding the 2D convolution filters and pooling kernels to 3D, allowing the model to learn spatial-temporal features from videos. Thus, the typical square filters become cubic, with an $N \times N$ filter becoming one of size $N \times N \times N$. The pre-trained I3D model trained on the Kinetics dataset [17] achieved the state-of-the-art performance for action classification. In many previous works [10,39], feature extraction is performed using two streams of data (RGB and optical flow), with each stream processed by an independent I3D model. In our work, we only used the RGB stream of I3D instead of both streams to obtain a less complex feature extractor.

2.2. Action recognition models

Autism related behaviors are dynamic events, and last for a variable duration of time. Hence, it is necessary to capture temporal information to model these behaviors and accurately classify them. The deep features extracted using the approaches outlined in Section 2.1 are used as inputs to temporal models which are trained to classify actions. We select the following four prominent temporal deep learning approaches, the LSTM and three temporal convolutional network methods. A summary of these models and their strengths and weaknesses are presented in Table 2.

2.2.1. Long short-term memory (LSTM)

The LSTM [15] is a type of recurrent neural network (RNNs) that is capable of learning temporal relationships from time series data. Compared with traditional RNN models, LSTMs address the vanishing gradient problem when the temporal dimension becomes large. There are three logic gates (an input gate, an output gate, and a forget gate) in a LSTM cell that control the flow of information into and out of the cell. The input gate controls which information should be stored in the hidden memory, while the forget gate controls which information in the memory should be forgotten from the previous cell state. The output gate is responsible for determining which information should be output of the LSTM at a given timestep. In this work, we use an LSTM network with the configuration shown in Table 3.

Table 2
Pros and cons of selected action recognition methods.

Action Model	Pros	Cons
LSTM	- Has gates that control the flow of information in and out of the cell. - Low memory requirements. - Faster training.	- May suffer from the vanishing gradiens when the data sequence is very long. - May overfit if not properly regularized.
TCN	- Able to perform temporal mapping of very long sequences well due to its multiple layers of dilated convolutions (large receptive field). - Can handle variable-length sequences. - Does not suffer from vanishing/exploding gradients. - Able to make more accurate predictions. - Faster training.	- Higher memory requirements compared to LSTM.
MSTCN	- Achieves better results than TCN as it operates on full temporal resolution (no pooling layers are used). - Able to perform temporal mapping of very long sequences well due to its multiple layers of dilated convolutions (large receptive field). - Can handle variable-length sequences. - Does not suffer from vanishing/exploding gradients. - Multi-stage architecture provides an improved receptive field. - Faster training.	- Higher memory requirements compared to LSTM.
MSTCN++	- Similar capabilities to MSTCN, however, MSTCN++ achieves better results than MSTCN due to the dual dilated layer.	- Higher memory requirements compared to LSTM.

Table 3
Configuration of LSTM architecture.

Layers	Configuration
Layer1	Bidirectional LSTM layer with 512 hidden neurons
Layer2	Bidirectional LSTM layer with 512 hidden neurons
Layer3	Bidirectional LSTM layer with 512 hidden neurons
Layer4	Dense(128), with Relu activation function
Layer5	Dense(3), followed by softmax function

Table 4
Configuration of TCN architecture.

Layers	Configuration
Layer1	Dilation TCN layer with kernel size = 5, level size = 5
Layer2	Dense(256), with Relu activation function
Layer3	Dense(3), followed by log softmax function

2.2.2. Temporal convolutional networks (TCN)

The TCN [21] is a temporal model proposed for video-based action segmentation and detection tasks. Due to their capacity to learn temporal relations among feature sequences, TCNs have been widely used for various video-based problem domains that seek to obtain a single [40] or sequence of prediction outputs [41,42]. In this work, we adapt the TCN architecture introduced in [21] to recognize autism-related activities performed by children. TCNs are different to RNN models based on the following characteristics: 1) Causal convolution, meaning that the value returned by the model at the current step only depends on information from previous observations, and there is no information flow from the future to the past. 2) Dilated convolutions, which enables TCN models to achieve an exponentially large receptive field. That means the TCN model can have a large receptive field, even with few layers. 3) Residual connections, which have been shown to be an effective method to address vanishing gradient issues that exist in deep neural networks, and allow the network to pass information in a cross-layer manner. The details of the TCN configuration used in this work is shown in Table 4.

2.2.3. Multi-stage temporal convolutional network (MS-TCN)

Building upon the success of the TCN architecture introduced in [43], Farha and Gall [39] proposed a multi-stage temporal convolutional network for temporal action segmentation. The MS-TCN architecture is illustrated in Fig. 2. At each stage of MS-TCN, an initial prediction is generated and subsequently refined by the next stage. Each stage consists of a single-stage TCN model, composed of a 1×1 convolution layer followed by multiple dilated 1-dimensional convolutional layers. Drawing inspiration from [44],

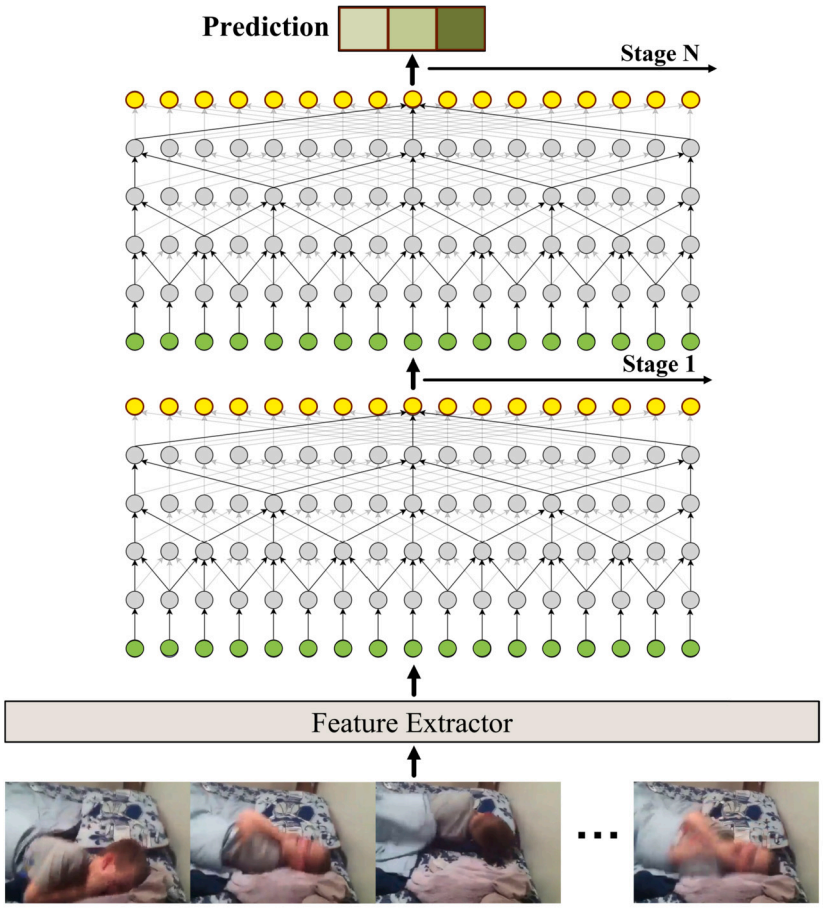

Fig. 2. Multi-stage temporal convolutional network. Each stage produces an initial prediction, which is refined by the subsequent stage. Recreated from [39].

Table 5
Configuration of MS-TCN architecture.

Layers	Configuration
Layer1	Dilation TCN layer with kernel size = 5, level size = 5
Layer2	Dilation TCN layer with kernel size = 5, level size = 5
Layer3	Dilation TCN layer with kernel size = 5, level size = 5
Layer4	Dilation TCN layer with kernel size = 5, level size = 5
Layer5	Dilation TCN layer with kernel size = 5, level size = 5
Layer6	Dense(256), with Relu activation function
Layer7	Dense(3), followed by log softmax function

the dilation layers are designed with a dilation factor that is doubled at each subsequent layer (*i.e.* 1, 2, 4, …, 512). This model does not use any pooling or fully connected layers in order to maintain the temporal resolution, and thus results in a lower trainable parameter count than other methods. We adopt the MS-TCN architecture for the behavior recognition task as shown in Fig. 2, and the details of the TCN configuration used in this work is shown in Table 5.

2.2.4. Extended multi-stage temporal convolutional network (MS-TCN++)

MS-TCN++ is an extension of the MS-TCN. The MS-TCN++ introduces a dual dilation layer to address the issue of smaller receptive fields in the lower layers, as compared to the MS-TCN. The dual dilation layer is depicted in Fig. 3. Similar to the MSTCN, the MS-TCN++ architecture is adopted here for the recognition of autism-related behavior. The details of the TCN configuration used in this work is shown in Table 6.

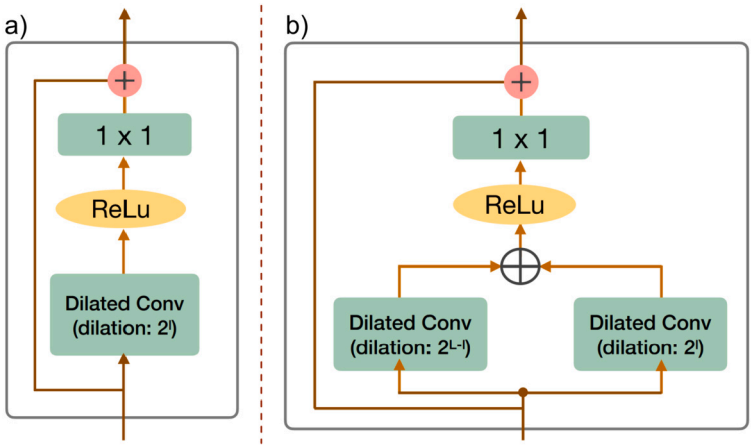

Fig. 3. A comparison between MS-TCN and MS-TCN++. (a) A presentation of the dilated residual layer in MS-TCN. (b) A dual dilation residual layer in MS-TCN++. Adapted from [45].

Table 6
Configuration of MS-TCN++ architecture.

Layers	Configuration
Layer1	Dual dilation TCN layer with kernel size = 5, level size = 5
Layer2	Dual dilation TCN layer with kernel size = 5, level size = 5
Layer3	Dual dilation TCN layer with kernel size = 5, level size = 5
Layer4	Dual dilation TCN layer with kernel size = 5, level size = 5
Layer5	Dual dilation TCN layer with kernel size = 5, level size = 5
Layer6	Dense(256), with Relu activation function
Layer7	Dense(3), followed by log softmax function

3. Experimental setup

3.1. Dataset and preprocessing

Due to ethical concerns, there are restrictions on the availability of public datasets that display autism behavior. To the best of our knowledge, the Self-Stimulatory Behavior Dataset (SSBD) dataset introduced by Rajagopalan et al. [12] is the only publicly available dataset. It should be mentioned that the dataset proposed by Negin et al. [18] is not currently accessible to the public. The complete set of videos reported in the SSBD dataset comprises 75 videos, of which only 60 are downloadable as the URLs are no longer available. The SSBD dataset contains three typical action classes: arm flapping, headbanging, and spinning. These categories of behaviors belong to the set of atypical motor behaviors [46]. Arm flapping is a repetitive behavior that is commonly observed when children are either unable to communicate effectively with those around them or when they have excess energy. Head banging, on the other hand, is a form of self-injury whereby children hit their heads with their own hands or against solid objects such as desks or walls. Finally, spinning is another common repetitive behavior observed in autistic children, where they rotate their entire body repeatedly. Selected samples are depicted in Fig. 4.

In order to evaluate the models, we conducted experiments where we carefully curated the SSBD dataset [12] and created a new dataset. The original SSBD dataset includes videos that were captured in very noisy and dark environments. We removed the noisy videos and supplemented the dataset with new videos collected from diverse sources, such as YouTube, Vimeo, and Dailymotion. These new videos were recorded in uncontrolled natural settings, showcasing interactions between parents and children, with both parties actively involved in prompting or playing with each other. Specifically, the videos were captured by parents in their daily living settings. The videos are closely reviewed and categorized into their corresponding autism-related behaviors, supported by information provided on the public repositories, and the repetitive actions that are a potential indication of ASD disorder are annotated. Furthermore, the dataset is not subject-oriented and is collected from different individuals. The final version of the dataset used in our experiments is composed of 61 videos and 61 subjects, and consists of 20, 21 and 20 videos for arm flapping, headbanging, and spinning, respectively. A comparison between the original and updated SSBD dataset is presented in Table 7.

For data preparation, we segmented videos into short clips to increase the number of samples. We fix the number of extracted key-frames from each sample video to 20 frames, which is sufficient to cover the complete length of each action. Raw frames from

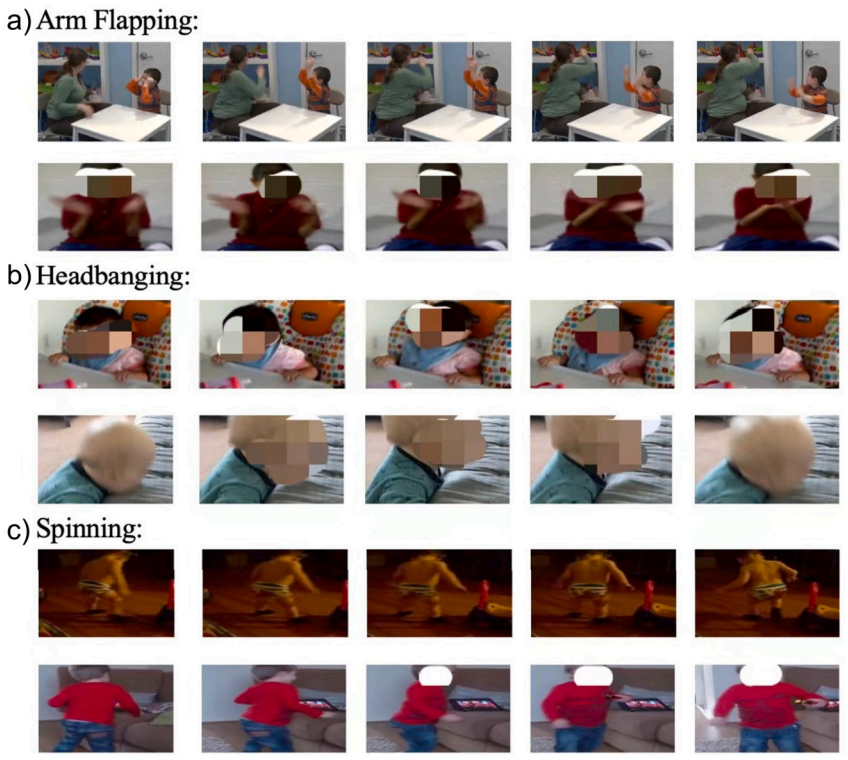

Fig. 4. Selected frames that illustrate the autism-related behaviors from the updated SSBD dataset. (a) Arm flapping (first two rows). (b) Headbanging (two middle rows). (c) Spinning (the last two rows). These videos were recorded in an uncontrolled environment, and some videos contain activities that involve human interactions (*e.g.* the sample in the first row).

Table 7
Comparison between the original and updated SSBD dataset.

	Armflapping	Headbanging	Spinning	Total
No. Videos in original SSBD dataset [12]	23	19	18	60
No. Videos in the curated SSBD dataset	20	21	20	**61**

Information of the source of data is available at Github Samwei1.

all video clips are extracted with FFmpeg.[3] After performing the data preparation, the total number of samples for each behavior in the ASD dataset were 57, 53, and 58 for arm flapping, headbanging and spinning, respectively.

The dataset is noisy and contains a large portion of the background or other subjects. To reduce the impact of the noisy environment captured within the data, we pre-processed the videos before training by detecting and cropping the target child from videos. In this way, irrelevant parts of videos are ignored, helping the proposed model analyze behaviors by learning more relevant visual features. For this purpose, we adopted the Mask-RCNN framework [48], which is a state-of-the-art object detector, to localize the target child and obtain a corresponding detection confidence score. We used the pre-trained weights obtained by training on the COCO dataset [49], and the implementation in Detectron2 [47]. Finally, we crop the raw images based on the detected bounding boxes, resize all frames to a fixed resolution based on the input size to the feature extractor, and pass these frames through the feature extractor. A representation of this pre-processing phase is depicted in Fig. 5.

We created an experimental dataset to improve the process of feature extraction and showcase the effectiveness of fine-tuning step of the feature extraction models on a subset of the training data, instead of solely relying on pre-trained feature extractors (*e.g.* pre-trained on ImageNet [33]). The Kinetics dataset [23] is a large action recognition dataset and contains some actions that are similar to the actions in our dataset. We combined action videos from Kinetics dataset with the training set videos from the updated SSBD dataset to construct a new dataset to help the feature extractors better understand the ASD behaviors. We named this dataset the *customized Kinetics dataset*.

[3] https://ffmpeg.org/.

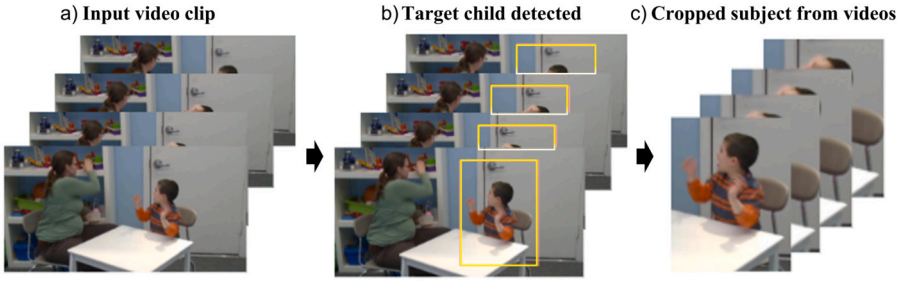

Fig. 5. The pre-processing procedure. (a) Input video clip. (b) The child of interest is detected using Detectron2 [47] human detector. (c) The target child is cropped from the video.

3.2. Evaluation protocol and implementation

In order to assess our proposed spatio-temporal methodologies, we employed 5-fold validation within a "Leave-One-Group-Out" setting, following the experimental design suggested in [12]. Consequently, each subject only appears in a single fold of the dataset, ensuring that the subjects used for training and testing in a specific fold are mutually exclusive.

As this is a multi-class classification task, the weighted F1-score is used as it considers each class's support and provides a more balanced view of the model's ability to correctly identify both positive and negative cases. We report on mean Weighted F1-score result for 5-fold validation. This metric can be calculated as follows:

$$\text{Weighted F1-score} = \sum_{i=1}^{3} \frac{2 \times precision_i \times recall_i}{precision_i + recall_i} \times w_i \tag{1}$$

where w_i is the weight of the i-th class, *i.e.* the percentage of the total samples that belong to the i-th class.

All proposed action recognition models introduced in Section 2.2 were trained from scratch using spatial features extracted from each frame by the trained feature extractor. The models underwent 200 epochs of training with a batch size of 16. The categorical cross-entropy loss was optimized using the Adam optimizer [50], employing a learning rate of $1e-3$ and decay rates of 0.9 and 0.999 for the first and second moments, respectively. Hyper-parameters were evaluated experimentally on the validation set, and the reported results reflect the parameters that yielded the highest accuracies during the cross-validation process.

4. Experimental results and discussion

We evaluated the performance of different combinations of features extractors and action recognition models for the detection of autism related behaviors using both light-weight and standard models. Due to the lack of publicly available code for related works [9,16–18,11,22], we compare our proposed methods with classical backbones (ResNet, I3D) and temporal models (LSTMs and TCNs) used for ASD action classification. The additional models explained in Subsection 2.1 and Subsection 2.2 are evaluated to verify performance in a low-computational resource setting, and to consider evaluate the impact of advanced temporal modeling methods. We note that unlike previous works in ASD action analysis, we are the first to propose a light-weight model system that can be used in future real-time applications.

4.1. Feature extractor evaluation

We investigated the performance of different feature extractors (light-weight and conventional) by comparing the performance of all feature extractors using a common TCN as the action recognition model. The light-weight feature extractors include MobileNet-V3, ShuffleNetV2, EfficientNet-B0, and ESNet. These are of more suitable, considering the ultimate objective of utilizing the models for real-world applications. The conventional feature extractors are ResNet18_vd, EfficientNet-B3 and the RGB stream I3D.

For the particular case of light-weight backbone models, we compare the performance of the models to extract relevant action features from video frames using the feature extractors already trained on ImageNet, and those fine-tuned with our customized Kinetics dataset as outlined in Section 3.1. Experimental results on two selected backbones are shown in Table 8. As in Table 8, the fine-tuning of backbones leads to a slight improvement in the accuracy of the action recognition model. Considering this improvement, we carried out all experiments on the light-weight backbones by first fine-tuning the feature extraction model.

All conventional feature extractors adopted were only pre-trained on the original Kinetics dataset, as we observed fine-tuning instability when using the modified Kinetics dataset due to the small size of the SSDB dataset and the complexity of the backbone, which resulted in catastrophic forgetting in some cases. The performance and the size of each feature extractor analyzed is shown in Table 9. According to Table 9, the performance of light-weight feature extractors is inferior to that of conventional and more complex ones. ESNet, among the light-weight models, attained the highest performance with a weighted F1-score of 0.63. Meanwhile, the RGB I3D backbone obtained the best weighted F1-score of 0.75 among the complex models.

Table 8
Five-fold evaluation results on the updated SSBD dataset to compare fine-tuned feature extractors.

Models	Weighted F1-score
ShuffleNetV2∗ + TCN	0.54
ShuffleNetV2⋄ + TCN	0.55
ESNet∗ + TCN	0.61
ESNet⋄ + TCN	0.63

∗ Pre-trained on ImageNet.
⋄ Fine-tuned on customized Kinetic dataset.

Table 9
Five-fold evaluation results on the updated SSBD dataset, using all feature extraction models and one traditional learning strategy TCN. The number of parameters for each model is also reported.

Backbone	Weighted F1-score	Parameters
MobileNet-V3⋆ + TCN	0.50	5.4M
ShuffleNetV2⋆ + TCN	0.55	1.17M
EfficientNet-B0⋆ + TCN	0.59	5.3M
ESNet⋆ + TCN	**0.63**	2.1M
ResNet18_vd† + TCN	0.52	11.72M
EfficientNet-B3† + TCN	0.67	12M
RGB I3D† + TCN	**0.75**	12.7M

⋆ Pre-trained on customized Kinetics dataset.
† Pre-trained on Kinetics dataset.

Model	Backbone	Weighted F1-score
LSTM	ESNet	0.63
	RGB I3D	0.71
TCN	ESNet	0.63
	RGB I3D	0.75
MS-TCN++	ESNet	0.66
	RGB I3D	0.78
MS-TCN	ESNet	**0.71**
	RGB I3D	**0.83**

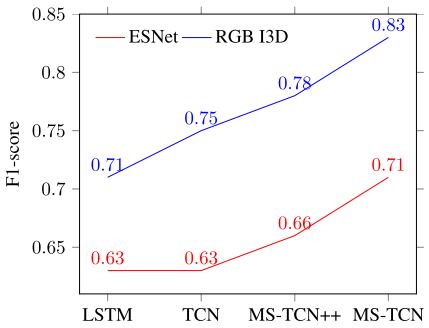

Fig. 6. Comparison between learning strategies with the proposed backbone networks for both light-weight and standard operating environments.

4.2. Action recognition models evaluation

Having determined the most appropriate feature extractors, our next objective is to evaluate the effectiveness of various action recognition models. In this regard, we employ the top-performing light-weight (ESNet) and conventional (RGB I3D) feature extractors as identified in Section 4.1. As outlined in Section 2.2, the action recognition models we consider are LSTM, TCN, MS-TCN and MS-TCN++. A comparison of the models is shown in Fig. 6. From Fig. 6, we can see the performance of the TCNs models are better than the LSTM for the ADS action recognition. This result also supports the conclusion of [21], where TCNs were shown to outperform recurrent neural networks such as LSTM for several tasks. The reason for this performance gain is that TCNs can have an effective longer memory than RNNs. In addition, the I3D+ and ESNet+ MS-TCN model achieved the highest weighted F1-scores of 0.83 and 0.71 on average for conventional and light-weight models, respectively.

In addition to the above experiments, considering the lack of availability of publicly available dataset and code for the ASD classification works, we adopted relevant methods used for ASD classification from [12,9,16,10] on our curated SSBD dataset and presented the results in Table 10. Our results demonstrate that MS-TCN could be a better option in comparison to other temporal networks.

It is noted that there is a trade-off between accuracy and computation time. With respect to computation times, ESNet+ MS-TCN model is the best solution because due to its lower complexity (around 2M parameters), while still achieving competitive performance (0.71 weighted F1-score). However, in terms of accuracy, the I3D+ MS-TCN model outperforms other all considered models in this work for the action recognition task (0.83 Weighted F1-score), yet has higher computational demands.

While the feature extraction component plays a crucial part in achieving a low number of floating-point operations per second (FLOPs), the adaptation of the action recognition model itself to real-world applications poses challenges. This is primarily due to the temporal aspect, which necessitates the provision of a sequence of frames. Thus, real-time performance of these methods is also

Table 10
Five-fold evaluation results on the updated SSBD dataset compared to frameworks used in the literature.

Models	Weighted F1-score
Harris3D detector + STIP (based on [12])	0.48
VGG16 + LSTM (based on [9])	0.64
Time Distributed CNN + LSTM (based on [16]	0.71
RGB I3D + LSTM (based on [10])	0.71
RGB I3D + MS-TCN (this work)	**0.83**

A comparison between our proposed pipeline and adopted ASD classification methods. Please note that to be fare in our comparison to these works we did not include Light-weight feature extraction backbones.

impacted by how the buffering is performed and the size of the frame buffer used for action recognition. With the proposed backbone and action recognition model, we aim to achieve the best trade-off between accuracy and latency.

We propose the following strategies: i) Instead of processing images one by one in a video sequence, processing image frames in a batch-wise manner can improve the computational efficiency. ii) To enable parallel computing in our system, we adopt a first-in-first-out (FIFO) buffer, where the data written into the buffer first comes out of it first. The feature extracted with EsNet is appended to a buffer of size 50, which operates in a sliding window fashion. The content of the buffer is finally sent to the trained MS-TCN model for action identification. Testing on a single Nvidia Geforce RTX 2080Ti and 4 CPU cores, the pipeline takes about 175 ms to process 50 frames of data. Although our results are promising, additional optimization and model pruning techniques are required for the action recognition model [45] (such as optimizing the computation graph generated in Pytorch with TensorRT) [51] to deploy our pipeline on an edge-AI system that can be built around, for example, a single embedded Jetson AGX Xavier device [52].

5. Conclusion

In this paper, we introduce a region-based computer vision system that aims to help clinicians and parents analyze children's behaviors, and in particular help to identify behaviors associated with Autism Spectrum Disorder (ASD).

We enhanced an existing dataset to construct a new, expanded dataset of stereotypical children's autism behaviors, using videos recorded in an uncontrolled environment and available from different online portals. We investigated several region-based action recognition frameworks for the detection of these characteristic behaviors, which incorporate light-weight and standard complexity feature extractors. The experimental results show that the proposed framework can recognize autism-related behaviors accurately and robustly. Further, from the preliminary experimental results, the light-weight backbone combined with the proposed temporal model demonstrated competitive results in comparison to the conventional model suitable for a powerful workstation. We verify the efficacy of the proposed strategy for potential real-time processing. In the future, we plan to explore more resource-efficient techniques to speed up the autism diagnosis process, and extend our system to other motor and mental disorders.

Compliance with ethical standards

The experimental procedures involving human subjects described in this paper were approved by the CSIRO Health and Medical Human Research Ethics Committee (CHMHREC). The CHMHREC is an NHMRC Registered Human Research Ethics Committee (EC00187). CSIRO Ethics ID 2022_004_LR.

CRediT authorship contribution statement

Pengbo Wei: Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper. **David Ahmedt-Aristizabal:** Conceived and designed the experiments; Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper. **Harshala Gammulle, Mohammad Ali Armin:** Conceived and designed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data; Wrote the paper.

Simon Denman: Analyzed and interpreted the data; Wrote the paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data associated with this study has been deposited at https://github.com/Samwei1/autism-related-behavior.

References

[1] D. Ahmedt-Aristizabal, S. Denman, K. Nguyen, S. Sridharan, S. Dionisio, C. Fookes, Understanding patients' behavior: vision-based analysis of seizure disorders, IEEE J. Biomed. Health Inform. 23 (6) (2019) 2583–2591.
[2] C. Lord, M. Elsabbagh, G. Baird, J. Veenstra-Vanderweele, Autism spectrum disorder, Lancet 392 (10146) (2018) 508–520.
[3] N.R. Council, et al., Educating Children with Autism, National Academies Press, 2001.
[4] Australian Institute of Health and Welfare, Autism in Australia, https://www.aihw.gov.au/reports/disability/autism-in-australia/contents/about.
[5] L. Crane, J.W. Chester, L. Goddard, L.A. Henry, E. Hill, Experiences of autism diagnosis: a survey of over 1000 parents in the United Kingdom, Autism 20 (2) (2016) 153–162.
[6] M. Del Coco, M. Leo, P. Carcagni, P. Spagnolo, P. Luigi Mazzeo, M. Bernava, F. Marino, G. Pioggia, C. Distante, A computer vision based approach for understanding emotional involvements in children with autism spectrum disorders, in: Proc. Int. Conf. Comput. Vis., 2017, pp. 1401–1407.
[7] R.A.J. de Belen, T. Bednarz, A. Sowmya, D. Del Favero, Computer vision in autism spectrum disorder research: a systematic review of published studies from 2009 to 2019, Transl. Psychiatry 10 (1) (2020) 1–20.
[8] A. Zunino, P. Morerio, A. Cavallo, C. Ansuini, J. Podda, F. Battaglia, E. Veneselli, C. Becchio, V. Murino, Video gesture analysis for autism spectrum disorder detection, in: Int. Conf. Pattern Recognit., IEEE, 2018, pp. 3421–3426.
[9] N. Kojovic, S. Natraj, S.P. Mohanty, T. Maillart, M. Schaer, Using 2d video-based pose estimation for automated prediction of autism spectrum disorders in young children, Sci. Rep. 11 (1) (2021) 1–10.
[10] A. Ali, F. Negin, F. Bremond, S. Thümmler, Video-based behavior understanding of children for objective diagnosis of autism, in: Proc. Int. Conf. Comput. Vis. Theory Appl., 2022.
[11] S. Liang, A.Q.M. Sabri, F. Alnajjar, C.K. Loo, Autism spectrum self-stimulatory behaviors classification using explainable temporal coherency deep features and svm classifier, IEEE Access 9 (2021) 34264–34275.
[12] S. Rajagopalan, A. Dhall, R. Goecke, Self-stimulatory behaviours in the wild for autism diagnosis, in: Proc. Int. Conf. Comput. Vis., 2013, pp. 755–761.
[13] S.S. Rajagopalan, R. Goecke, Detecting self-stimulatory behaviours for autism diagnosis, in: Proc. Int. Conf. Image Process., 2014, pp. 1470–1474.
[14] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint, arXiv:1409.1556, 2014.
[15] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[16] P. Washington, A. Kline, O.C. Mutlu, E. Leblanc, C. Hou, N. Stockham, K. Paskov, B. Chrisman, D. Wall, Activity recognition with moving cameras and few training examples: applications for detection of autism-related headbanging, in: Ext. Abstr. CHI Conf. Hum. Factors Comput. Syst., 2021, pp. 1–7.
[17] J. Carreira, A. Zisserman, Quo vadis, action recognition? A new model and the kinetics dataset, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 6299–6308.
[18] F. Negin, B. Ozyer, S. Agahian, S. Kacdioglu, G.T. Ozyer, Vision-assisted recognition of stereotype behaviors for early diagnosis of autism spectrum disorders, Neurocomputing 446 (2021) 145–155.
[19] P. Pandey, A. Prathosh, M. Kohli, J. Pritchard, Guided weak supervision for action recognition with scarce data to assess skills of children with autism, in: Proc. Conf. AAAI Artif. Intell., vol. 34, 2020, pp. 463–470.
[20] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. 25 (2012).
[21] S. Bai, J.Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv preprint, arXiv:1803.01271, 2018.
[22] X. Wang, Z. Miao, R. Zhang, S. Hao, I3d-lstm: a New Model for Human Action Recognition, in: IOP Conf. Ser. Mater. Sci. Eng., vol. 569, 2019, 032035.
[23] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al., The kinetics human action video dataset, arXiv preprint, arXiv:1705.06950, 2017.
[24] M. Tan, Q. Le, Efficientnet: rethinking model scaling for convolutional neural networks, in: Int. Conf, Mach. Learn, 2019, pp. 6105–6114.
[25] S. Xie, R. Girshick, P. Dollár, Z. Tu, K. He, Aggregated residual transformations for deep neural networks, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 1492–1500.
[26] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: efficient convolutional neural networks for mobile vision applications, arXiv preprint, arXiv:1704.04861, 2017.
[27] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, et al., Recent advances in convolutional neural networks, Pattern Recognit. 77 (2018) 354–377.
[28] N. Ma, X. Zhang, H.-T. Zheng, J. Sun, Shufflenet v2: practical guidelines for efficient cnn architecture design, in: Proc. Eur. Conf. Comput. Vis., 2018, pp. 116–131.
[29] G. Yu, Q. Chang, W. Lv, C. Xu, C. Cui, W. Ji, Q. Dang, K. Deng, G. Wang, Y. Du, et al., PP-PicoDet: a better real-time object detector on mobile devices, arXiv preprint, arXiv:2111.00902, 2021.
[30] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, M. Li, Bag of tricks for image classification with convolutional neural networks, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 558–567.
[31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 2818–2826.
[32] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: CVPR, 2016, pp. 770–778.
[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 248–255.
[34] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al., Searching for mobilenetv3, in: Proc. Int. Conf. Comput. Vis., 2019, pp. 1314–1324.
[35] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: an extremely efficient convolutional neural network for mobile devices, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 6848–6856.
[36] C. Cui, T. Gao, S. Wei, Y. Du, R. Guo, S. Dong, B. Lu, Y. Zhou, X. Lv, Q. Liu, et al., PP-LCNet: a lightweight cpu convolutional neural network, arXiv preprint, arXiv:2109.15099, 2021.
[37] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 7132–7141.
[38] K. Han, Y. Wang, Q. Tian, J. Guo, C. Xu, C. Xu, Ghostnet: more features from cheap operations, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 1580–1589.
[39] Y.A. Farha, J. Gall, Ms-tcn: multi-stage temporal convolutional network for action segmentation, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 3575–3584.
[40] H. Gammulle, T. Fernando, S. Sridharan, S. Denman, C. Fookes, Multi-slice net: a novel light weight framework for Covid-19 diagnosis, in: Proc. IEEE Int. Conf. Auton. Syst., IEEE, 2021, pp. 1–5.
[41] M.-H. Chen, B. Li, Y. Bao, G. AlRegib, Z. Kira, Action segmentation with joint self-supervised temporal domain adaptation, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 9454–9463.
[42] H. Gammulle, S. Denman, S. Sridharan, C. Fookes, Tmmf: temporal multi-modal fusion for single-stage continuous gesture recognition, IEEE Trans. Image Process. 30 (2021) 7689–7701.
[43] C. Lea, M.D. Flynn, R. Vidal, A. Reiter, G.D. Hager, Temporal convolutional networks for action segmentation and detection, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 156–165.

[44] A. v, d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, K. Kavukcuoglu, Wavenet: a generative model for raw audio, arXiv preprint, arXiv:1609.03499, 2016.
[45] H. Gammulle, D. Ahmedt-Aristizabal, S. Denman, L. Tychsen-Smith, L. Petersson, C. Fookes, Continuous human action recognition for human-machine interaction: a review, arXiv preprint, arXiv:2202.13096, 2022.
[46] S.E. Bryson, L. Zwaigenbaum, C. McDermott, V. Rombough, J. Brian, The autism observation scale for infants: scale development and reliability data, J. Autism Dev. Disord. 38 (2008) 731–738.
[47] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, Detectron2, https://github.com/facebookresearch/detectron2, 2019.
[48] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proc. Int. Conf. Comput. Vis., 2017, pp. 2961–2969.
[49] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: common objects in context, in: Proc. Eur. Conf. Comput. Vis., 2014, pp. 740–755.
[50] D.P. Kingma, J. Ba Adam, A method for stochastic optimization, arXiv preprint, arXiv:1412.6980, 2014.
[51] NVIDIA, Tensorrt, https://github.com/NVIDIA/TensorRT, 2019.
[52] D. Franklin, jetson-inference, https://github.com/dusty-nv/jetson-inference, 2019.