Video-Based Activity Recognition for Child Behaviour Understanding

Thesis submitted to the University of Utrecht for the degree of
MSc, July 2021.

Feyisayo Olalere

6702481

Supervised by

Metehan Doyran, MSc.
Dr. Ronald Poppe
Prof.Dr. Ali Salah

Abstract

Over the years, deep learning models have been able to record state-of-the-art (SOTA) performance on the task of activity recognition. The results of this can be seen in applications such as video surveillance, medical diagnosis, robotics for human behavior characterization, and like in this study, recognition of human activities from videos. One of the factors that have contributed to the benchmark performance of these models is the availability of large-scale datasets. However, we have observed that these datasets are largely skewed towards adults. That is, they contain more videos of adults than kids. Out of 5014 videos from an adult-specific dataset, only 1109 videos contained kids performing an action. Since there exist visual differences in how an adult performs an activity as opposed to a child, in this study, we test if current SOTA deep learning models have some systemic biases in decoding the activity being performed by an adult or a kid. To do this, we create kid-specific and adult-specific datasets. Using a SOTA deep learning model trained on the different datasets, we test for the generalization ability of the deep learning model. Our results indicate that, while SOTA deep learning models can be used to classify kid activities, the kid-specific dataset is more complex to generalize to than the adult-specific dataset. The study also shows that the features learned from training on a kid-specific dataset alone can be used to classify adult activities while the reverse is not the case.

Acknowledgements

I will like to thank my supervisory team: Ronald, Metehan, and Albert. It really took a village to make this work. Thank you, Alex, for willingly providing your models and answering all my questions (there were a lot!). A very big thank you to my project team member, Vincent!

Most importantly, I acknowledge that it took the grace of God and my loving family and friends to be able to do this. A big thank you to Sam (in all 525 languages).

Contents

Chapter 1

Introduction

1.1 Motivation

Activity recognition for human behavior analysis is one of the major problems being researched within the computer vision community [Aggarwal and Xia (2014)]. The increase of interest in this research area is fueled by the availability of more datasets, increased hardware complexity, advanced computer vision techniques, and the need for various applications in the real world [Borges et al. (2013)]. These applications include video surveillance systems, robotics for human behavior characterization, medical diagnosis, and many more [Vrigkas et al. (2015), Poppe (2010)]. While the field has seen more advancement over the decade, there remain areas where activity recognition has not been fully applied. One of such areas is activity recognition for children's behavioral analysis. More research into this area will bring about better onset diagnosis of children related diseases such as cerebral palsy or diseases that affect the neuromotor development of children [Hesse et al. (2018); Chambers et al. (2020)]. It could also help to understand children's' behavior better, and to develop interactive

playgrounds.

To create the applications mentioned above, we need to automatically detect and analyze the behavioral cues that give an insight into the psychological state, emotional state, cognitive state of a person, and many more. The application of computer vision techniques to behavioral analyses works by recognizing non-verbal cues such as facial expressions, body gestures, body poses, and other visual cues in videos and images [Vrigkas et al. (2015)]. Since these non-verbal cues are made up of combinations of complex actions, to identify them, we must first successfully identify these complex actions. The field of activity recognition applies state of the art (SOTA) deep learning models to detect these actions. By the application of SOTA deep learning models to children's data, we can identify these actions and in turn identify and analyze the behaviors defined by these actions.

1.1.1 Activity Recognition

The research area of activity recognition is a subfield of computer vision that has to do with identifying and classifying different activities in real-life settings [Kim et al. (2009)]. While they are some standing challenges such as occlusion, differences in how multiple people perform the same action, the application of SOTA deep learning models to current human activities datasets has significantly improved the accuracy at which machines can successfully predict human activities.

An activity is a set of actions performed consecutively with a certain pattern [Poppe (2010)]. For an activity class like soccer game, the actions will include kicking a ball, dribbling, etc.

However, these SOTA deep learning models have majorly been applied to

adult-specific benchmark datasets rather than kid-specific data. One of the reasons for this is the unavailability of an adequate benchmark dataset for children's activity recognition.

Kid-specific dataset in this study, refers to large activity datasets containing more videos of children performing an activity than footage with adults performing the same activity.

Adult-specific dataset refers to current benchmark datasets for activity recognition which have more videos of adults performing an activity and less videos of kids performing the same activity. Examples include; Kinetics [Smaira et al. (2020)], Charades [Sigurdsson et al. (2016)] and UCF101 [Soomro et al. (2012)].

Activity Recognition For Children

There is a significant variation in how people (adults and children) perform actions. However, we can't say if there is a systemic bias in how the behavior of children differs from adults. In this study, we explicitly investigate this problem.

With current SOTA deep learning models, a lot of research has been done in detecting and analyzing behavioral attributes by recognizing adult activities. By carrying out this study, we address the question, can these same models be used for kid-specific activity recognition.

1.2 Research Objectives

To guide this study, the following research question will be answered:

Main Research Question: Do the current state of the art (SOTA), deep learning models, for activity recognition generalize to kids-specific dataset?

To answer the main research question, we need to break it down into the following research questions.

- **RQ1:** Can an Adult SOTA model generalize to a Kid-specific dataset?

- **RQ2:** Can a Kid SOTA model generalize to an Adult-specific dataset?
- **RQ3:** Does training on both kids and adults-specific dataset increase the performance of the model on:
    a. Adult-specific dataset?
    b. Kid-specific dataset?
    c. Does increasing the size of training data in the mixed model improves the model's generalization to the kid-specific dataset ?
    d. Does increasing the size of training data in the mixed model improves the model's generalization to the adult-specific dataset

In RQ1, we examine how a SOTA deep learning model trained on an adult-specific dataset performs on a kid-specific dataset. The performance analysis will be based on the errors and biases this model encounters when evaluated on a kid-specific dataset. To answer RQ2, we will train a kid-specific SOTA deep learning model and see how it performs when evaluated against the adult-specific dataset. In RQ3, we will check how the mixed model performs on kids-specific data and how it does on adult-specific data.

The results gotten here will be compared to all the previously ran experiments to see, which data type the model needs to generalize more to kids' data.

By answering these research questions, we will be able to determine if current adult-specific SOTA deep learning models can be used to build kid-specific activity recognition systems and how much value is added to activity recognition models when kid-specific data are included in training the model

1.3 Thesis Contribution

To the best of our knowledge, most of the studies that have been conducted regarding activity recognition for children make use of non-visual data [Boughorbel et al. (2010); Nam and Park (2013); Ahmadi et al. (2018); Suzuki et al. (2012)]. While the few that provide visual kid-specific dataset have relatively small dataset [Rajagopalan et al. (2013)], some of which were captured in monotonous environments [Hesse et al. (2017); Hesse et al. (2018)]. Hence, this is the first study conducted about analyzing the use of SOTA deep learning models for kids' activity recognition with sufficiently large kid-specific activity data. The following contributions will be made by this study:

1. We will create a dataset for kid activity recognition.
2. We will perform a detailed quantitative and qualitative analysis on the use of SOTA for kid activity recognition

1.4 Thesis Outline

In Chapter 2, we discuss related works and methodologies relevant to our main research question. In Chapter 3, we discuss our data collection pipeline. In Chapter 4, we discuss the methodology applied in this study. In chapter 5 we present and discuss the results of our experiments. In chapter 6, we discuss the limitations and future work. We present our conclusion in Chapter 7.

Chapter 2

Literature Review

2.1 Computer Vision

Computer vision is a sub-field of artificial intelligence that addresses the problem of how computers can "see" and "understand" the contents of an image or video. Earlier studies approached this problem by handcrafting features that are representative of the different parts of the images [Norvig and Intelligence (2002)]. However, this required a large deal of manual labor and domain expertise. Over the years, as computational power and knowledge about the field increased, machine learning techniques that can automatically identify these features and process them have been developed. One of which is the biologically inspired convolutional neural networks (CNNs). This technique is inspired by the human vision and has brought about advancement within computer vision and other subfields of artificial intelligence.

2.1.1 Convolutional Neural Network (CNN)

A CNN is a feedforward multilayered network with three major operations at each layer, convolution, activation function, and pooling. The convolution operation extracts useful features from the image by convolving a filter over different parts of the image and computing the dot product. The output from this is passed into an activation function which learns different abstractions and introduces non-linearity into the feature space. In most cases, subsampling is performed on the output of the activation function through the pooling operation. This reduces the computational requirement of the network by summarizing what has been learned. Also, the operation makes the network invariant to geometrical changes in the input [Khan et al. (2020)].

Other modifications applied to CNNs, such as increasing their depth, hyperparameter tuning, and training on large-scale datasets have led to their improvements at various vision tasks.

2.1.2 Transfer Learning

Traditionally, CNNs trained on a particular domain's data can only be used for tasks in that domain. For example, a model trained using only images of dogs would have a reduced performance if it was asked to recognize cats. This is a problem because it means for every new domain we need to build a large enough dataset and train the CNN from scratch. Building a dataset and training a CNN are both computationally and manually expensive. The main idea behind transfer learning comes from the notion of humans using previous knowledge in a particular domain to infer new things in a similar domain. Hence, transfer learning makes CNNs pre-trained in one

domain reusable for a similar domain. For example, the features learned by CNN used to identify bicycles can be used to identify scooters by fine-tuning it on a smaller scooter dataset instead of training from scratch.[Lu et al. (2015)].

Transfer learning is useful when the source domain is different from the target domain but has some similarities. If the label information of both the source and target domain is known, the process is referred to as *inductive transfer learning* as with our case. Pan and Yang (2009) present four approaches to transfer learning.

Instance-transfer approach: In this method, there is the assumption that if all of the data in the source domain can not be used, they are certain parts of the source data that can be re-weighted and reused in the target domain. Re-weighting and important sampling are two major techniques used for this approach. [Jebara (2004); Jiang and Zhai (2007); Liao et al. (2005)].

Feature representation transfer approach: The idea is to learn "good" features that reduce the differences between the source and target domain. So the knowledge transferred to the target domain is encoded in the feature representations. These feature representations are expected to lead to significant improvement in performance in the target domain. These features can be constructed using supervised or unsupervised learning approaches depending on the availability of data labels in the source domain. [Argyriou et al. (2007); Jiang and Zhai (2007); Lee et al. (2007)].

Parameter-transfer approach: The assumption here is that there are shared parameters or priors between the models of the source and target domain. Hence, knowledge from the source domain can be encoded in the model's parameters or priors and transferred to the target domain.

[Lawrence and Platt (2004); Evgeniou and Pontil (2004); Gao et al. (2008)].

Relational knowledge transfer approach: This approach assumes there is some relationship in the data of the source and target domain. Hence, the knowledge transferred is the relationship among the data. [Mihalkova et al. (2007); Mihalkova and Mooney (2008); Davis and Domingos (2009)].

2.2 Human Activity Analysis

It has been observed that literature makes use of the term "activity" and "action" interchangeably. While we already define that we consider an activity to be a longer temporal sequence, we would also make use of the term interchangeably here. This is to avoid confusion with the literature cited as they use the term action for what we consider an activity. So, every occurrence of the word action in this section is used for an activity.

Based on Aggarwal and Ryoo (2011) categorization of human activities (gestures, actions, interactions and group activities), Lei et al. (2019) classify human activity analysis into 3 problems; action recognition, action prediction and action quality evaluation.

The task of action quality evaluation [Pirsiavash et al. (2014); Morel et al. (2016)] involves assessing how an action was performed and providing semantically correct feedback on how to improve the action. An example application is a physiotherapy system that accesses how the person performs a required action and provides feedback on what the subject should do better. For action prediction [Kong and Fu (2017); Hu et al. (2018)], the aim is to determine the action label based on partially observed or

incomplete actions that occurred in the input video. With both tasks, temporal information can be obtained from RGB videos or depth videos. Unlike action recognition, still-frames and images are hardly used for both tasks. Lastly, the action recognition task [Herath et al. (2017); Ziaeefard and Bergevin (2015)] aims to determine what action is occurring and when does the action occur. Two of the typical problems with this task are action classification and action detection. In the former, the aim is to determine the action label of the given image or video, while the latter aims at determining the beginning and end of an activity in a given video. [Lei et al. (2019)]

In the next section, we discuss the different methods for performing activity recognition as this forms the basis of our work in children's activity recognition.

2.3 Methods for Activity Recognition

The first question that is posed with activity recognition is, *"how do we represent each activity in a video"* [Kong and Fu (2018)]. This question is particularly challenging because of factors such as variations in how the activity is performed and dynamic background (see Section 2.4 for the challenges). The task of activity representation is to convert the activity in the video into a feature vector that can be used to infer the activity label of the given video. [Wang et al. (2016a); Li et al. (2017); Kong and Fu (2018)]

A feature is a descriptive, informative, and discriminative representation of an activity in a video. It is also referred to as a descriptor.

Based on numerous studies, activity representation is categorized into two methods; handcrafted feature methods and (deep)learning-based methods [Herath et al. (2017); Zhang et al. (2019); Lei et al. (2019)]. In the former, the features representing the activity are manually defined as opposed to deep learning methods where these features are automatically learned.

In the following sections, we start by discussing hand-crated feature representation (global and local representation methods). After this, we present deep-learning feature representation methods.

2.3.1 Hand-crafted Feature Representation

This representation method is further categorized into two methods; global representation and local representation. Global representation employs a top-down approach where the subject is first localized and then the regions of interest are encoded as a whole. While local representation follows a bottom-up approach where the point of interest is first detected, then local patches are calculated around these points. These patches are combined to form the final representation. Based on extensive surveys [Poppe (2010); Herath et al. (2017); Kong and Fu (2018)] on action recognition, we explain these methods below.

Global Representation

With global representation, activity recognition is dependent on successfully encoding the full representation of the subject performing the activity. These representations can be obtained from the subject's silhouette, contour, or optical flow. However, in capturing the information within the region of interest, these representations become susceptible to noise, view-

point variations, background clutters, and partial occlusions [Herath et al. (2017); Poppe (2010)].

One of the first work that uses silhouettes is by Bobick and Davis (2001). They present two templates for representing motion in frames using a single image. The first template, Motion Energy Image (MEI) shows *where* the motion occurs in the frame, while the second template, Motion History Image (MHI) shows *how* motion moves in the frame (see Figure 2.1). These templates capture relevant contextual information within the videos and can be used for tasks such as filtering out cluttered backgrounds in images [Tian et al. (2011)]. While these templates lead to an improvement in the result of recognition systems, the templates are variant to changes in viewpoints.

To address the viewpoint sensitivity problem with MEI and MHI, Weinland et al. (2006) creates a new view-invariant motion descriptor called Motion History Volumes (MHV). Unlike [Bobick and Davis (2001)], they use multiple calibrated video cameras to get silhouettes from different viewpoints and combine them into a 3D voxel model. In addition to extending the 2D templates to a 3D template, they also make use of Fourier transform and cylindrical coordinates along the medial axis to make their template location and rotation invariant. Wang et al. (2007) applies Radon transform to silhouettes to encode low-level features that are invariant to transformation and robust to noise. Chen et al. (2006) makes use of contour descriptors. They describe human posture using star skeletons which connect the center of an object to the contour extremes.

Instead of using shape-based (silhouettes or contours) properties for activity representation, motion information can also be used. One of the common motion information is optical flow [Horn and Schunck (1981); Sun

et al. (2010)]. Optical flow estimates the motion field from videos by mapping the pixels from one frame to another. It is calculated from pairs of subsequent frames. Unlike shape-based representations, optical flows can be used when background subtraction cannot be done. In Efros et al. (2003), they make use of sport footages to calculate optical flow. The flow field was then split into four distinct channels to capture both the vertical and horizontal motions across the frames (see Figure 2.5). This method was used by Ali and Shah (2008) to obtain a set of kinematic features from optical flow and by Wang and Mori (2010) to generate features that describe the human body parts.

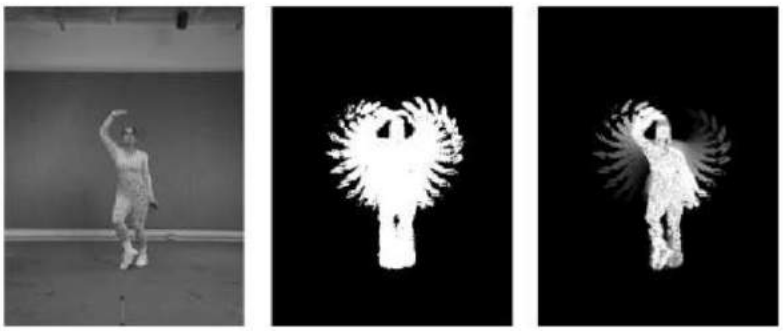

Figure 2.1: An example from [Bobick and Davis (2001)] showing the input video frame, the extracted motion energy image, and the motion history image (from left to right).

While global representation was favored in earlier research on activity recognition, the focus has now shifted to local and self-learned representations. Limitations such as rigidity of the method to caption all variation that could occur with activity and inability to capture more fine-grained details are attributed to this shift [Dollár et al. (2005); Matikainen et al. (2009); Herath et al. (2017)].

Figure 2.2: Original frame

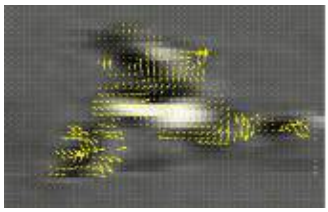

Figure 2.3: Optical flow

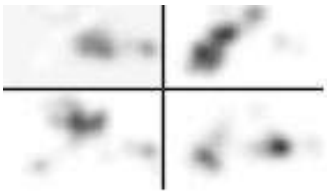

Figure 2.4: Flow field in four channels

Figure 2.5: Example from Efros et al. (2003) showing the optical flow and the field in four channels based on the original frame

Local Representation based Methods

Local representation summarizes the activities in an image or video using a collection of sampled local descriptors. Different from global representation, there is no need to first localize the person(s) performing the activity or do background subtraction. Ideally, local descriptors should overcome the problems of global descriptors such as occlusions, variance to viewpoints, and personal appearance. In applying local representation to activity recognition, the region of interest is first detected, and then the local descriptors are extracted and aggregated to form the representation of that activity [Herath et al. (2017)].

Space-time interest point (STIP) is a method to detect the region of interests based on the space-time region. The motion information within these regions tends to be more informative and more noticeable than its surrounding regions. Laptev (2005) built a STIP detector by extending the Harris corner detector [Harris et al. (1988)] to a 3D version. The 2D- Harris corner detector detects interest points that have significant changes in

two orthogonal directions. In extension, the 3D approach identifies points where the image values have a significant variation in both space and time dimensions. Willems et al. (2008) also extends a 2D interest point detector (Hessian detector) into a 3D version. Contrary to using gradients to detect these points as was done by [Laptev (2005)], they make use of second-order derivatives to find these points. A common limitation to the 3D-Harris and 3D-Hessian detector in certain domains like facial expression recognition is that the spatiotemporal corners needed by these detectors rarely occurs. While attributes like sparseness could help the recognition model generalize better, too much rarity could make it difficult for recognition [Dollár et al. (2005)]. To address this limitation, Dollár et al. (2005) explicitly define their detector to select too many features rather than too few. They apply a 2D Gaussian smoothing kernel along the spatial dimension, a quadrature pair of 1D Gabour filter along the temporal dimension and fine-tune their detector to select spatiotemporal regions where the variations in image intensity evoke complex motions. Apart from STIP-based methods, dense sampling is another way to pinpoint relevant regions in the frame. Wang et al. (2009) compared the performance of Laptev (2005), Willems et al. (2008) and Harris et al. (1988) on a task of human action recognition and found that dense sampling outperformed their STIP detectors.

After the spatiotemporal regions are detected, local descriptors are extracted within these regions. In early works like [Dollár et al. (2005); Laptev (2005)], obtaining the local descriptor around space-time interest point was done using cuboids, 2D cuboids for images, and 3D-cuboids for videos. Later studies [Messing et al. (2009); Matikainen et al. (2009); Wang and Schmid (2013); Wang et al. (2013)] introduced the notion of motion trajectories as an alternative means to obtaining local descriptors around the points of interest.

Histogram based descriptors are commonly used local descriptors for edges and motion [Herath et al. (2017)]. Klaser et al. (2008) propose the use of a 3D gradient orientation descriptor. To create this descriptor, they extend the concepts of Histogram of Gradient (HoG) descriptors applied to static images [Lowe (2004); Dalal et al. (2006)] to 3D (HoG3D). This is to capture the spatiotemporal volumes in videos. Other studies like 3D SIFT [Scovanner et al. (2007)] and local trinary pattern [Yeffet and Wolf (2009)] also extend existing 2D descriptors into their 3D versions. Laptev et al. (2008) suggest the use of both Histogram of Optical Flow (HoF) descriptors and HoG descriptors computed within the neighborhood of the interesting point. HoF can draw on the property of optical flow field that allows for encoding motion at a pixel level across the frames. Dalal et al. (2006) introduced the Motion Boundary Histogram (MBH) which also uses the key concepts of HoG but computed on motion boundary fields. These fields are obtained by taking the gradients of the optical flow field which also allows them to compress camera motions. Other local descriptors include spatiotemporal version of local binary pattern (LBP) [Ojala et al. (2002); Zhao and Pietikainen (2007); Kellokumpu et al. (2008)] and SURF features [Willems et al. (2008)]. These descriptors successfully make use of cuboids in extracting their local descriptors.

Another way of extracting local descriptors from interest points is with the use of motion trajectory. A trajectory is a feature that is tracked over time. They are particularly useful for features with a long temporal extent. In Sun et al. (2009), they extract trajectories by computing pairwise matching over the detected SIFT points that occur across consecutive frames. To mitigate the creation of spurious trajectories, they impose a unique match constraint and leave out trajectories that occur too far apart. In Wang and Schmid (2013), they extract the HoG, HoF, and MBH at each trajectory

and combined them to form an improved dense trajectory (IDT). They also improve the performance of dense trajectory by correcting for camera motion estimated with the homography between consecutive frames. Jiang et al. (2012) also show that correcting trajectories using camera motion leads to improvements in performance.

Usually, the number of descriptors extracted across videos tends to vary and have high dimensionality, and cannot be directly compared. Hence, we need a way to aggregate them into distinctive and fixed-size feature vectors that can be learned from and eventually compared against new videos. One of the popular approaches to doing this is based on the concepts Bag-of-Words for text categorization.

Bag-of-Visual Words (BoV) [Csurka et al. (2004); Herath et al. (2017)] is a histogram of codeword occurences in the "codebook" or "visual vocabulary". The codebook consists of clusters of local descriptors and the middle or the closest descriptor is selected as the codeword. Even though studies like [Montoliu et al. (2015); Boufama et al. (2017)] used BoV methods for activity recognition, BoV does not capture temporal information in its original form. Laptev et al. (2008) propose spatiotemporal grids as a way to retain some spatial information. Other improvements to the original BoV includes [Kovashka and Grauman (2010); Liu et al. (2011)].

Fisher Vector (FV) [Perronnin and Dance (2007)] is another method for aggregating local descriptors. This an extension to the BoV approach and is based on the principles of Fisher Kernels [Jaakkola et al. (1999)]. The main idea here is to represent the input image with a gradient vector obtained from a generative probability model and then feed the vector through a discriminative model. When a Gaussian Mixture Model is used for feature generation, FV can benefit from both first and second-order statistics

during aggregation. However, this results in FV's having high dimensions [Jégou et al. (2010)]. To reduce the dimensionality, Jégou et al. (2010) introduces Vector of Locally Aggregated Descriptor (VLAD) which removes the second-order information from the descriptor and has about half of the dimensionality of FV. By reducing the dimension, they did not compromise accuracy as their performance outperformed other SOTA and was comparable to the result produced by the BoV approach. In Perronnin and Larlus (2015), they combined FV with CNNs. In the first layer of the architecture, they apply an unsupervised approach to local feature extraction, computing the FV encoding and reducing its dimensionality. Then they combined this with a regular CNN trained with backpropagation. At the end of the study, they were able to show that the mid-level feature extracted by their architecture was on par with those derived by a full CNN.

Other methods for descriptor aggregation includes, the use of dictionaries [Guha and Ward (2011); Sadanand and Corso (2012)], and Hidden Markov Models [Hongeng and Nevatia (2003); Tang et al. (2012)].

2.3.2 Deep-Learning Feature Representation

Despite the advancements handcrafted features (global and local) has brought to the field of activity recognition, these methods require a great deal of manual labor and knowledge about the task domain. These are some of the reasons why the shift is being made toward feature representation using deep learning techniques [Simonyan and Zisserman (2014); Liu et al. (2016); Zhang et al. (2018)]. Convolutional neural networks (CNNs) are one of the best deep-learning algorithms for exploring spatial contents in videos and images, and current SOTA deep-learning models are largely based on CNNs. These models differ in terms of whether a 2D or 3D convolution is

used, what is passed in as input (RGB or optical flow), and what information is propagated through the network [Carreira and Zisserman (2017)]. They are continuously being used as a baseline for both feature extraction and recognition and have been shown to perform better than SOTA handcrafted feature extractors like HoG [Everingham et al. (2010)]. We present some of the recent SOTA deep-learning algorithms following the category as presented in [Kong and Fu (2018)].

Spatiotemporal networks: CNNs can extract and aggregate features in a given frame through the convolution operation. Currently we can have a 2D convolution [Karpathy et al. (2014)] or 3D convolution [Tran et al. (2015)] operation. The difference is that the former can only extract features in the spatial dimension and would require some additional aggregation method to capture motion or temporal information. However, the latter considers both spatial and temporal dimensions when extracting these features (see Figure 2.6). It is more intuitive to make use of the 2D CNN in detecting the features in images or a single frame at a time, however, since we have multiple frames in a video, it would be useful to capture the temporal information over time. The use of 3D kernels for convolution and 3D pooling is a way to retain temporal information.

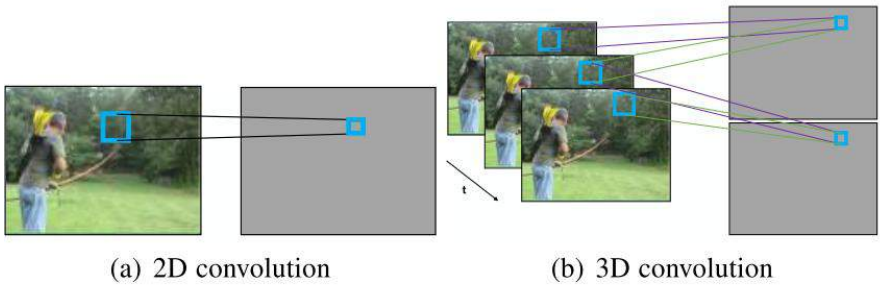

Figure 2.6: An example from [Kong and Fu (2018)] illustrating the 2D and 3D convolution operation.(from left to right)

Ji et al. (2012) introduce a 3D CNN architecture that makes use of 3D convolution. This was achieved by convolving a 3D kernel over adjacent

input frames. Instead of randomly initializing their model, they start with 5 hardwired kernels namely, gray, gradient-x, gradient-y, optflow-x, and optflow- y. After getting the resultant feature map from the first layer, they repeat 3D convolution and subsampling across each channel, and then a fully-connected layer was used to generate the final feature vector for action classification (see Figure 2.7). The model was able to achieve competitive performance on the TRECVID and KTH datasets.

Tran et al. (2015) extends the 3D ConvNet into a deep architecture (C3D) trained on a large-scale dataset (Sports-1M). Their architecture maintained a homogenous temporal depth where the kernel in each convolution layer is the same size (3 x 3). C3D was able to learn better feature embedding for video than other methods (see Figure 2.8). The study also showed that the learned C3D features with a linear SVM can outperform or approach other SOTA approaches on the benchmark data.

While 3D convolution has proved useful in learning temporal information from videos, they have more parameters than 2D ConvNets which makes them harder to train. Also, they seem to not be able to benefit from reusing ConvNets pre-trained on large-scale image datasets. Carreira and Zisserman (2017) propose two-stream inflated 3D ConvNets (I3D) which can leverage 2D ConvNets pre-trained on large-scale image data. They inflate the 2D ConvNet to 3D by adding a temporal dimension to the filters, and they also bootstrap the parameters learned from the 2D ConvNet. This architecture outperforms other SOTA models for activity recognition on the Kinetics-400 dataset. Also, by pre-training on the kinetics dataset, the method achieves high performance on UCF101 and HMDB51 datasets. Tran et al. (2018) proposed a new spatiotemporal block (R(2+1)D) where they factorize the 3D convolution into two operations, a 2D spatial convolution and a 1D temporal convolution. This approach produced SOTA

performance on the large-scale dataset and performs better than the I3D model at action recognition. Further studies like Ghadiyaram et al. (2019) leverage on the R(2+1)D architecture for pretraining, and they achieve stellar performance on the kinetic-400 dataset. In this study, we also make use of this architecture as our baseline model.

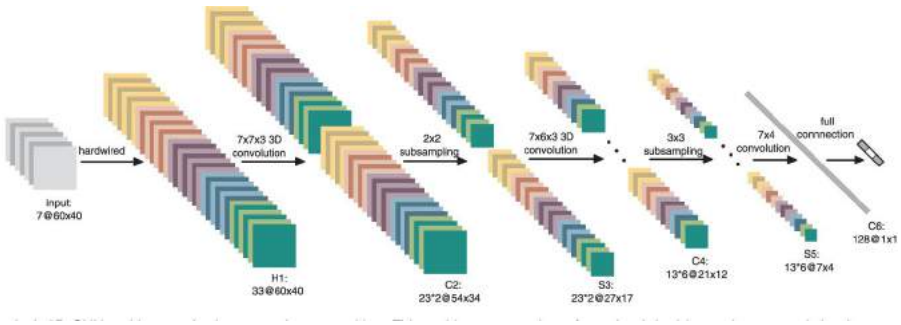

Figure 2.7: 3D ConvNet for human action recognition created in [Ji et al. (2012)].

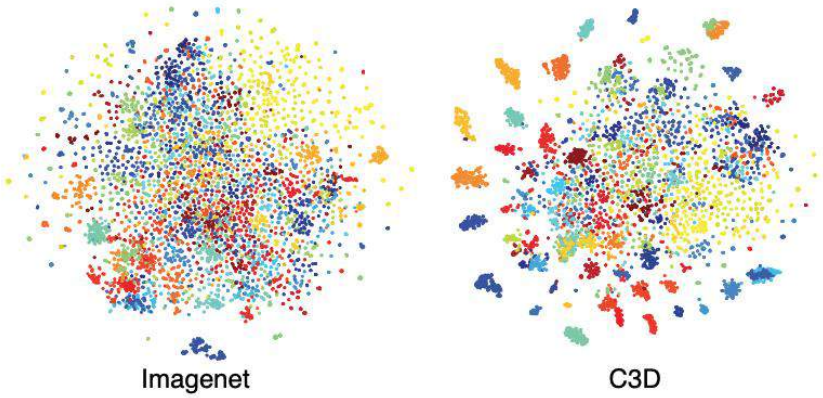

Figure 2.8: Video feature embedding by Imagenet and C3D. We can see clearer clusters with C3D than with imagenet, which shows C3D is better at learning features for videos. Originally shown in [Tran et al. (2015)].

Multi-stream networks: The idea behind multi-stream networks is to model temporal and spatial information using a spatial ConvNet that takes in still images and a temporal ConvNet that takes in motion information from the optical field. The output from each CNN is fused at some specified convolution layer. Earlier work by Simonyan and Zisserman (2014) directly combine the output generated by the softmax layer at the last layer of the network (see Figure 2.9). However, to model spatiotemporal information,

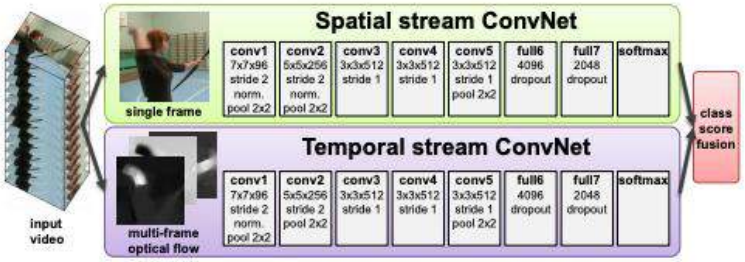

Figure 2.9: Two-stream architecture proposed in [Simonyan and Zisserman (2014)]. The spatial stream models takes in static image as input and the temporal stream takes in optical flow from consecutive frames.Originally shown in [Simonyan and Zisserman (2014)].

earlier interaction between the two streams could also be useful. Feichtenhofer et al. (2016) perform their first fusing after an intermediate convolution layer, and this improved their performance and also significantly reduced the training parameters required. Their result showed that the best accuracy is obtained by fusing after the last convolution layer. Other more recent structures based on multi-stream architecture have been proposed. Wang et al. (2016b) proposed Temporal Segment Networks (TSN). They increase performance by extracting short video segments through a sparse sampling approach, and then they aggregate the information learned from the snippet. Their approach can capture long temporal information in the video. In Feichtenhofer et al. (2016), they combine ResNets with a two-stream CNN by connecting residual connections to the pathways between the two streams. Later in Carreira and Zisserman (2017), they make use of a 3D ConvNet in their two-stream architecture. Although 3D convolution can capture motion information, their results show that adding extra motion information from the optical field improves performance at the recognition task. Sarabu and Santra (2020) reduces the number of redundant features generated by their two-stream architecture by training the spatial stream with a ResNet and the temporal stream with Inception-v2.

Hybrid networks: Hybrid networks aggregate temporal information by

adding a recurrent network on top of 2D CNN architectures [Wang et al. (2014); Yue-Hei Ng et al. (2015)]. This type of networks leverage on the advantages of both CNNs and LSTMs and have proven to be able to capture long-range dependencies and spatiotemporal information [Wang et al. (2015); Kar et al. (2017); Diba et al. (2017)]. Donahue et al. (2015) propose the use of Long-term Recurrent Convolutional Networks (LRCNs) which combines 2D CNN and LSTM architecture for video activity recognition amongst other tasks (see Figure 2.10). Their study showed that by stacking LSTM's on top of 2D CNNs, they can capture temporal dependencies that can not be captured by just the 2D CNN. In Yue-Hei Ng et al. (2015) they propose two CNN based methods that could learn from full-length videos. The first method makes use of temporal feature pooling for feature aggregation (they try 6 methods) and the second feeds the input from the CNN into LSTM layers. Their results showed that using LSTM layers on top of the CNN architecture outperforms the temporal pooling method by a small margin. Wu et al. (2015) stacks bi-directional LSTM(BiLSTM) model on a two-stream CNN architecture. The two-stream CNN extracts motion information from the input and then feeds this into the BiLSTM to model long term temporal dependencies.

2.4 Challenges of Deep-learning Models for Human Activity Recognition

Although major advances have been made in activity recognition, there remain some challenges that make SOTA algorithms prone to errors in real-world scenarios. [Kong and Fu (2018)]

Intra-class and inter-class variations: In an ideal situation, we want

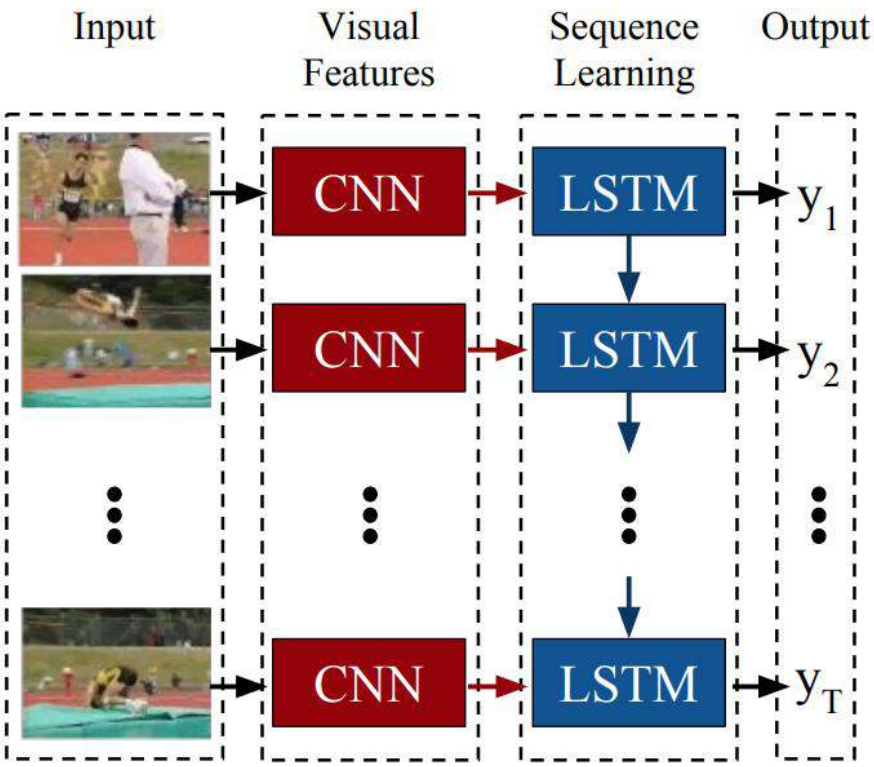

Figure 2.10: Donahue et al. (2015) learn temporal information by sequentially feeding the output from the CNN into the stacks of LSTM layer. Originally shown in [Donahue et al. (2015)].

the dataset to exhibit small intra-class variation and a large inter-class variation, however this is not so in real-world settings [Li et al. (2020); Akila and Chitrakala (2018)]. For one, the way people perform the same activity can differ. For example, with an activity like running, people run at a different pace (fast, slow, etc), and people's posture while running can differ. Apart from this, these activities could be captured at different viewpoints, front view, side view, overhead view, etc. These factors lead to large intra-class variability within the training and test data, which makes SOTA prone to misclassification. Furthermore, multiple activities share close similarities. For example, activities "sailing on a boat" vs "sailing on a yacht". Such inter-class similarities also lead to misclassification of activities by SOTA algorithms.

Cluttered background and environmental factors: SOTA algorithms

also draw features from the backgrounds in the activity frame [Laptev et al. (2008); Dollár et al. (2005)]. While this would not be a problem if the data was gathered in a controlled environment with little variation in the background, this is problematic in real-world settings. The background noise the systems encounter in a natural environment can degrade the recognition performance of the algorithms. However, backgrounds can also provide useful features for improving activity recognition. For example, diving can be characterized by the presence of water or golf by the presence of green open space [Tu et al. (2018)]. Other environmental factors such as shadows, lightning conditions, color variation, and occlusions are part of the challenges that affect recognition algorithms [Poppe (2010); Kong and Fu (2018)].

Temporal variations: The extent of temporal features that can be extracted from the videos is one of the differences between using videos as opposed to images for activity recognition. However, this temporality is also a challenge. For one, activities in the dataset might have been performed and recorded at different rates, hence, the recognition algorithms have to be invariant to the different rates in the dataset. Moreover, deep neural architectures currently cannot effectively deal with temporal variations. The SlowFast [Feichtenhofer et al. (2019)] architecture address this problem a bit.

Static vs moving cameras: Videos from static cameras are relatively easier to process because the background is non-changing and, motion is only observed with the moving object or subject in the frame. However, challenges like occlusion, illumination, or difficult background subtraction because the subject is wearing a similar color to that of the background, still occurs. As opposed to static cameras, the video obtained from moving cameras can capture a larger extent of motion but they are more difficult

to process since both the background and foreground appear moving. In addition to the challenges mentioned with static cameras, moving cameras can introduce blurs to the video due to abrupt change of scene [Chapel and Bouwmans (2020)].

Obtaining and labeling data: While the web now provides a pool from which datasets can be constructed, this is still quite a challenging task. Firstly, not all the data gathered is suitable for the task. This means that after preprocessing, we are left with fewer data for training than was gathered. Secondly, since the data gathered are raw and unlabelled, labeling has to be done either manually or automatically. Due to the volume of the data gathered, it is a time-consuming effort to manually annotate them. Also, with manual annotations we run into problems with inter-annotator agreements [Patron-Perez et al. (2007)]. Even though there are automatic methods that make use of metadata such as subtitles [Gupta and Mooney (2009)], search results [Schroff et al. (2010)] to label the data, it still requires a manual check to see if the right label was assigned to the right activity. Although an unsupervised learning approach would require no label to learn, we can't guarantee that semantically meaningfully classes would be learned [Poppe (2010)]. The active learning approach has been applied to reduce the time required by manual annotation [Ahmadi et al. (2018)].

Redundancy and uneven predictability: In video data, not all frames are useful in recognizing the activity done, which means the video could contain a lot of redundant frames. While a small group of frames can be sufficient in defining some activities, other activities require more frames to define them. This results in uneven predictability of the activity classes. Also, since the frames with the required context-information can occur anywhere in the video, there is the challenge of predicting an activity as

early as possible in real-world applications. Studies like Gupta and Mooney (2009) re-arranged their frames such that the frames with relevant context-information appear early in the video, but the performance of the algorithm was still limited due to insufficient relevant frames in the video.

All the issues mentioned above have to be explicitly addressed when building an activity recognition system.

Chapter 3

Data Collection

To answer the research questions defined in Section 1.2, we need a sufficiently large kid-specific dataset to train deep-learning models. However, we observed that the currently available benchmark activity recognition datasets are adult-specific, i.e most of the videos contain adults performing the activity. Hence, to facilitate this research, we created a kid-specific dataset called Kinetic-kids (see Fig 3.1).

3.1 Data Download

The kinetic-kids currently contain clips of kids within the ages 0-12 (prepubescent) performing 21 sporting activities (see Appendix A). The classes in this dataset are derived from the Kinetics-400 dataset [Kay et al. (2017)]. While the Kinetics-400 dataset has over 38 sporting classes, not all the sport classes are commonly performed by kids in our age group, hence there are not enough youtube videos available for kids performing the sport classes excluded. We choose to use the classes from the Kinetics-400 dataset because of the extensive works that have been done using this dataset. Fur-

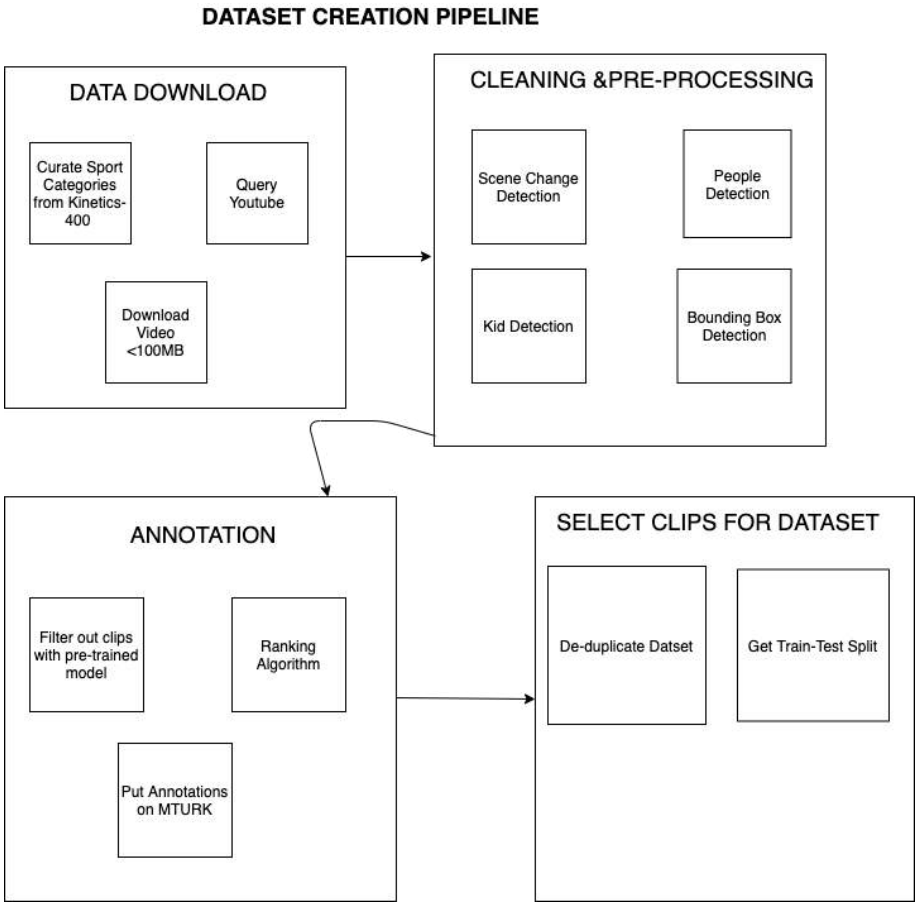

Figure 3.1: Data Collection Pipeline

thermore, a number of high-performing SOTA activity recognition models have been trained using this dataset [Tran et al. (2018), Feichtenhofer et al. (2019)]. In addition to these reasons, we are able to extract an adult-specific version of kinetics by having annotators select which videos have adults performing the activity. This allows for a comparison of how the SOTA activity recognition model performs on kid vs adult-specific datasets.

3.1.1 Creating Queries and Downloading Videos

We first compile a list of sports activities based on kinetics-400 to include in this dataset (38 activities). After an extensive filtering process (see Fig 3.1), we selected 21 sport activity labels to include in our dataset.

We choose the sports category based on our hypothesis that an observable difference should exist between how a sporting action is performed by a kid as opposed to an adult (see Fig 3.3). We work with only sporting categories in this project because of time and resource constraints.

Figure 3.2: From the left kids bowling, hitting baseball and playing tennis

Figure 3.3: From the left adults bowling, hitting baseball and playing tennis

After compiling the activity labels for Kinetic-kids, we define specific query lists tailored to search for videos with kids performing these activities. We tailor our query to target the age group we are interested in and search for them on Youtube. For example, *basketball game in pre-school* or *kids dunking basketball*. Before downloading the returned videos, we check that the video is at most 100MB. We do this to filter out professionally shot and heavily edited videos, also to reduce the resources required for storage. The non-professionally shot videos are less edited and are more depicting of the real world. We also check that we have good enough resolutions, the maximum resolution we collect is 720p videos. Hence, the videos in our dataset can also be used for estimating kids' poses. The videos that meet both criteria are downloaded and saved for pre-processing and annotation.

3.2 Cleaning & Pre-processing

After the videos have been downloaded, we start pre-processing by performing a scene detection step on each video. Our reason behind this step is, we observed that actions that span across two scenes are usually different and nothing is linking the motion from one scene to another. Since there is no motion information needed for classification during a scene change, we reduce the memory requirement and speed during pre-processing by running the rest of the steps on scenes rather than the full video.

Using *PySceneDetect* [Castellano (2017)], we check for differences in the HSV space between consecutive frames. If the difference in average HSV pixel values between consecutive frames is greater than 30%, we presume a scene change. We start by splitting the videos into scenes as we deem that actions that appear across the scene boundaries are not informative for us.

3.2.1 People Detection

After performing scene detection for each video, we select 3 evenly spaced frames from scenes that are longer than 1 second and pass these through a pre-trained YOLO-V3 [Redmon and Farhadi (2018)] model for people detection. This detection model is reported to have a person detection mAP of 50.3% and was chosen due to its excellent speed-accuracy quotient. Its accuracy is less than the 56.4% mAP that the Faster R-CNN model employed by Simple Baseline and HRNet achieves, but requires only a fraction of the computational cost. We take this step to eliminate scenes that do not have people in them, as as we are only concerned with scenes containing human activities. Detections with confidence of less than 70% are discarded.

3.2.2 Child detection

Since our end goal is to end up with a kid-specific dataset, we filter out scenes that do not contain children. Most current age recognition techniques are limited to faces (via datasets such as MORPH [Ricanek and Tesafaye (2006)], CACD [Chen et al. (2014)] and FG-NET [Fu et al. (2014)], AFAD [Niu et al. (2016)], and [UTKFace Zhang et al. (2017)]) or to voice recognition. For our purpose, however, we cannot rely on facial features as this would induce an obvious bias towards front-facing subjects, nor can we rely on voice features as possible speech in the video does not have to originate from our subject. Work exists that distinguishes children from adults on anatomical differences by use of their different head-to-body ratios [Ince et al. (2014, 2017)], though these models are not public and the authors ignored our access requests. To our knowledge, there is no other work published to specifically identify children based on full-body cues that work from various angles and require no specialized hardware (such as 3D cameras [Basaran et al. (2014)]).

We instead detect children using the recent zero-shot model CLIP [Radford et al. (2021)], developed by OpenAI. Traditional image recognition models are trained in a supervised manner on a hand-crafted dataset to predict the probability of an image belonging to one or more labels out of a set. CLIP is instead trained on 400 million automatically collected image-text pairs and outputs cosine similarities between pairings. By comparing the similarities of an image to several hand-crafted indicator sentences, we can map an image to the class it is most similar to.

To test the CLIP model for child detection, we annotated a subset of our downloaded data. This gave us 1001 adult and 349 child images. After annotation, we perform an ablation study to determine which parameters

give better detection of children.

First, we developed prompts describing our images, as was done in the CLIP's study. Our prompts included sentences such as *a photo of a teenager, a photo of a child*. We added a margin of 0.2 to the images because the bounding box detected by using the YOLO model in the previous step was sometimes too small. Also, we apply padding to the images to make them square-shaped as CLIP only takes as input, square-shaped images. We padded by mirroring the image along the edges of the image (reflect padding). With these parameters, we got the lowest AP and AUC from the model (see 3.1).

Using the same margin scale and padding method, we optimize the way we feed our input to the model by using the same prompts used in training CLIPs on ImageNet. This gave an increase in CLIPs performance over when we only use the prompts developed by us. Finally, we append the word *doing ¡sports¿* to all the prompts used in training CLIPs on ImageNet (ImageNet+sports). Here *¡sports¿* is replaced with one of our sporting categories. This means the prompts (text) in the image-text passed into CLIP looked like *An image of a child doing badminton*. Using these prompts gave us the highest AUC and AP in this phase of our ablation.

We carried out a second phase of the ablation study. In this phase we used the ImageNet+sports prompts, reflect padding method and, we varied the margin scale applied to the images. From this phase, we realized that a margin scale of 0.2 works better for the model to classify kids. In the final phase of the ablation, we use the ImageNet+sports prompt, a margin scale of 0.2 and, we vary the padding method used. We tried zero-padding where the edges of the images are set to zero (black), one-padding (set the edges to 1), and reflection padding where we repeat the last pixel of the image

across the edges of the image.

Based on the result of the ablation study, our final model configuration uses ImageNet+sports prompts, a margin scale of 0.2, and zero-padding (See results in 3.1).

	Prompt type	Margin scale	Padding	AP	AUC
	"a photo of label"	0.2	Reflect	0.689	0.421
	ImageNet	0.2	Reflect	0.712	.443
	ImageNet + sports	0.2	Reflect	0.769	0.434
	ImageNet + sports	0	Reflect	0.750	0.460
	ImageNet + sports	0.1	Reflect	0.765	0.453
	ImageNet + sports	0.3	Reflect	0.765	0.411
*	ImageNet + sports	0.2	Zero-Padded	0.813	0.468
	ImageNet + sports	0.2	Replication	0.804	0.441
	ImageNet + sports	0.2	One-Padded	0.814	0.458

Table 3.1: Ablation study of hyper parameters for our CLIP child detector. The row indicated with a * shows our final configuration

To fine-tune the predictions of the model such that it differentiates between prepubescent children and teens, we also use an ensemble of indicator labels per class. The labels "infant", "toddler", "child" all indicate our desired "child" class, whereas "adult" is indicated by "adult" and "teen". Instead of cosine similarities per label, we want to have the model output a single value in the range of $[0, 1]$ as our child probability. Formalized in equation 3.1, we calculate this by taking the cosine similarities Z and pick the maximum cosine similarities $\mathcal{Z}$ of the labels for both our "child" and "adult" classes. We convert these into probabilities via a softmax step σ. Finally, since this is a 2-class problem, it suffices to just use the probability of our

"child" class.

$$
\begin{aligned}
Z &= CLIP(x,\ \text{labels},\ \text{prompt templates}) \\
\mathcal{Z}_0 &= \max(\{z \in Z \mid z \text{ is child label}\}) \\
\mathcal{Z}_1 &= \max(\{z \in Z \mid z \text{ is adult label}\}) \\
P &= \sigma(\mathcal{Z}) \\
P_{child} &= P_0
\end{aligned}
\tag{3.1}
$$

In summary, we run our zero-shot child detector once for each of our detected YOLOV3 bounding boxes. We take a crop of the image for each bounding box, with a margin of 20% in both the width and the height (chosen via the ablation experiments in Table 3.1). As CLIP accepts only 224x224 images, we rescale the crops such that their longest side fits these restrictions size and zero-pad them to fill the square.

3.3 Annotation

The pre-processing step filters out the scenes in each video where no kid is detected (See Fig 3.4). At the end of that phase, we end up with videos containing just scenes with a confidence level of at least 70% that a child exists in the scene.

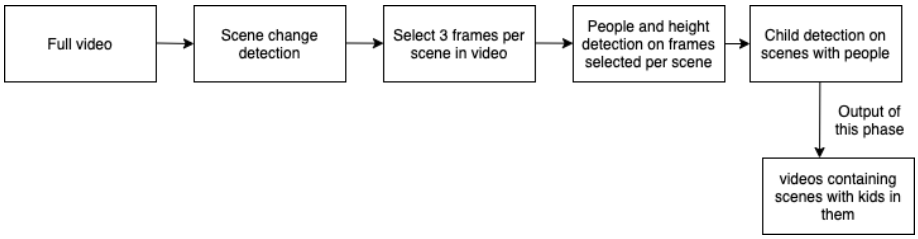

Figure 3.4: Cleaning and Prepossessing Pipeline

To create a robust dataset that presents some difficulty to current SOTA

deep learning models, we perform an extra filtering step. We classify the activities present in each scene per video with a SlowFast model [Feichtenhofer et al. (2019)] trained on the kinetics-400 dataset [Guo et al. (2020)]. The SlowFast model reads input through 2 branches. The fast branch requires more frames than the slow branch. For the fast branch, we pass as input 32 frames selected from the initial 64 frames (stride=2) and, we pass an input of 4 frames into the slow branch (stride=16). The scenes that do not have up to the required number of frames are filtered out because their duration is too short (¡2s) and not so useful for us.

At this step, we save the top-5 predictions made by the model for each scene in a video alongside their prediction probability. The prediction probability is used by the ranking algorithm to select which scenes per video contain activities that could be possibly difficult to current SOTA models. The ranking algorithm is presented in the following sub-section.

3.3.1 Ranking Algorithm

Based on the model's prediction, we know the top-5 labels predicted by the model and with what certainty this prediction was made. The ranking algorithm checks for the scenes in a video where the model was most uncertain about predicting the correct class label. At this phase, a correct label means that the model predicts the actual label of the video e.g *hitting baseball* or predicts a label that falls within the super-class of a category. For example, the basketball category contains labels, *shooting basketball, dunking basketball*, etc. If the model makes any of these predictions for a video within this category, we count this as a correct category. This is because we only split videos into their respective classes after manual labeling (explained in 3.3.2).

Scene Number	1	2	3
Label 1	opening present	skipping	**dribbling basketball**
Prediction-Probability 1	0.321	0.301	0.401
Label 2	**playing basketball**	crying	catching or throwing baseball
Prediction-Probability 2	0.212	0.145	0.03
Label 3	hitting baseball	**dunking basketball**	unboxing
Prediction-Probability 3	0.156	0.01	0.003
Label 4	somersaulting	egg hunting	catching or throwing softball
Prediction-Probability 4	0.132	0.001	0.002
Label 5	playing tennis	doing laundry	playing cricket
Prediction-Probability 5	1.101	0.0001	0.001

Table 3.2: Example of result saved when a video containing 3 scenes is evaluate with the SlowFast model (video falls under the basketball category)

We will use the results in Table 3.2 to explain the ranking algorithm. The ranking algorithm starts by checking which super-category the video evaluated belongs to. In the example above, the video falls under the *basketball category*. Hence, the algorithm is only interested in finding scenes that have one of the subclass labels predicted by the model. In this case, the sub-class labels are; [*playing basketball, shooting a basketball, dribbling basketball and dunking basketball*]. The goal of the algorithm is to create a clip of at most 60 seconds that contains the scenes where the activity was most uncertain.

From Table 3.2, the algorithm will rank scene 2 with the highest score because the label dunking basketball (Label 3) was predicted with 0.01, which is the least probability made by the model across all the scenes that had a basketball sub-label predicted. Scene 2 is then added as the start of the new clip. Next, the model checks if the duration of the new clip is greater than 60 seconds. If it is, the algorithm stops, and the new clip containing only scene 2 is presented for manual labeling. If it is less than 60 seconds, the algorithm checks if a scene number lower or higher than the already picked scene has a basketball sub-label predicted for it. This means we check scene 1 and scene 3. In this case, both scenes 1 and 3 have a basketball sub-label predicted in their top-5. So we pick the scene where

the basketball sub-label has a lower prediction probability. That is scene 1 (Label 2 and probability 0.212). Scene 1 is then added to the new clip at the correct temporal position. So the new clip now contains [scene 1 and then scene 2]. We check again if the new clip is > 60 seconds and repeat the process, till all the scenes containing the basketball sub-label have been added to the new clip or stop once the duration exceeds 60 seconds.

We sort the videos this way to maintain the temporal ordering in the video because they help provide more information about which activity is about to occur or is occurring. While the majority of the clips at this processing step are less than or equal to 60 seconds, we do have some clips that are more than 60 seconds. This happens when we have a single scene in a video and the duration for that scene is already more than 60 seconds (See Fig 3.5). We limited the duration to 60 seconds so the annotators do not have to watch long clips before making their decisions.

3.3.2 Manual Labelling on MTURK

Since most clips generated with a ranking algorithm had a low prediction probability, there is a need to manually confirm that each clip had a kid performing one of the activities of interest to us. For the manual annotation of the clips, we choose to use a crowd-sourcing platform, Amazon Mechanical Turk (AMT). AMT is very often used for tasks such as this one [Kay et al. (2017), Caba Heilbron et al. (2015)] and we expect that since the workers on this platform are more used to such tasks, it increases our chances of getting high-quality annotated data.

We customized the available template on AMT to fit our purpose (see Fig 3.7). The design choices with the interface were made to make the labeling

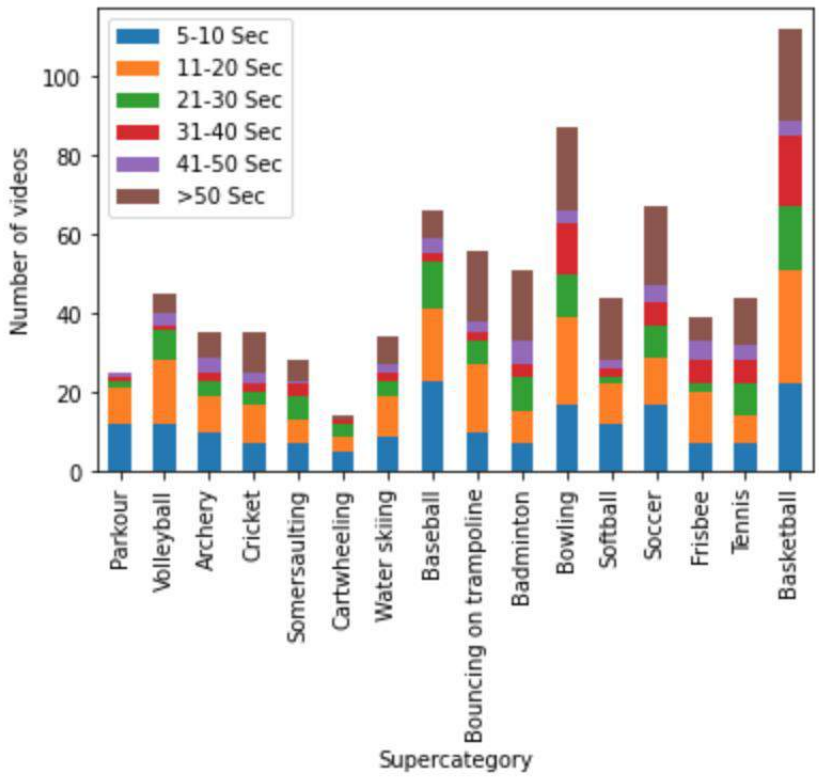

Figure 3.5: Duration distribution of videos presented for annotation on MTURK"

tasks as seamless as possible. In crafting the instructions, we took into consideration the visual differences between clips where a kid performs an activity versus where and how an adult would perform the same activity. For example, clips with a kid shooting a basketball vary in the environment it is performed (see Fig 3.8); in the living room, from the staircase, outside the house. Also how it is performed (see Fig 3.9); driving a cart to shoot the ball, standing on a platform, or jumping over some stool. We made sure to include examples that show the workers how different these activities could be performed so that they know what to annotate. The workers were presented with 5 activity labels and 2 other labels to choose from. If an activity label is selected, the workers are presented with functionality that helps them insert the time when the activity is starting. We also ask the workers to say if the activity is performed with an adult and if there are multiple instances of the action occurring in one clip. We asked

for the latter because we realized that some of the downloaded videos are compilation videos of distinct activity clips put into one clip and by knowing this, we will be able to cut more videos out for the dataset. The former provides a useful signal that can be used later for studying interactions between kids and adults (see Fig 3.6).

Figure 3.6: Adult perform activity with child. On the left, adult windsurfs with child, right, adult kicks soccer ball with child.

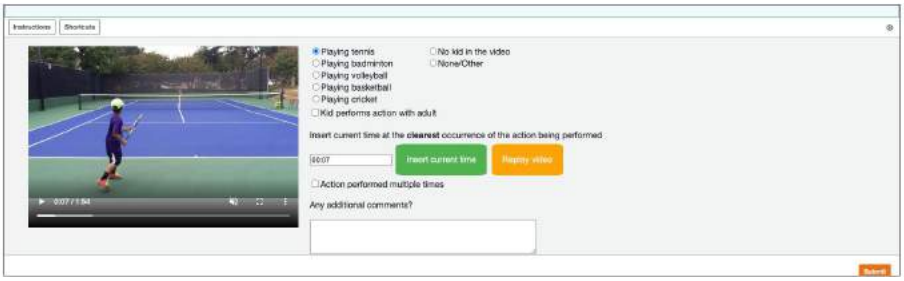

Figure 3.7: Labelling interface used in Amazon Mechanical Turk

Figure 3.8: Kids shoot basketball in three different environment.

Figure 3.9: Kids shoot basketball in three different ways.

We cluster similar sports together to form a super-category. Hence, workers on AMT get similar sports with each batch of videos they label. For

example, a super-category of *bat/racquet sports* contained videos of tennis, badminton, and cricket. Presenting workers with similar videos in a batch allows for faster acclimatization with the activities, helps improve the probability of selecting the right labels for each video, and helps us to contrast similar actions e.g *playing basketball* vs *dribbling basketball*.

The clips are presented to the annotators muted, to allow for concentration on the visual aspect of the activity. Each clip is annotated by two distinct workers. We checked to see if there is a trend between annotators' agreement and the duration of the clips presented for annotation but we could not observe a clear trend between the two variables. After annotation is done by two Mturkers,

1. we remove videos where both workers agree that there is no required activity in the video clip or the activity is not being performed by a kid.

2. If both workers agree on the label but disagree on the time:

    - If the difference between specified activity start times is less than 2 seconds, we take the average of the start time, sample a 5-second clip starting at the average time, and present it to a third annotator for confirmation.

    - If the difference in start time is more than 2 seconds, we present two 5 seconds sampled clips based on the activity start time entered by each annotator to a third annotator.

3. If both workers disagree on the label, we present two sampled 5-second clips based on the specified start times to a third annotator as two distinct clips with the label specified by each annotator for confirmation. This lead to cases where the same clip had different

labels and was added to the dataset. This kind of duplication is removed during de-duplication.

If the third annotator agrees with the annotation then we add the clips to our dataset. After processing and annotating all the clips that were downloaded by us, we ended up with 218 kid-specific videos. To increase the size of the dataset, we annotated the videos downloaded in the Kinetics-700 dataset for the categories we are interested in. Annotators were asked to select clips that have kids performing the action of interest. We annotated 5014 videos from the Kinetics dataset (Train and Val split), out of which we downloaded 1979 videos containing adults performing a sporting activity and 1109 containing kids performing the same activities (see Fig 3.10). We did not download all the videos because most of the videos annotated from kinetics were of adults. To handle class imbalance across both datasets by selecting 40 videos per class category in both datasets for training. (see Fig 3.10)

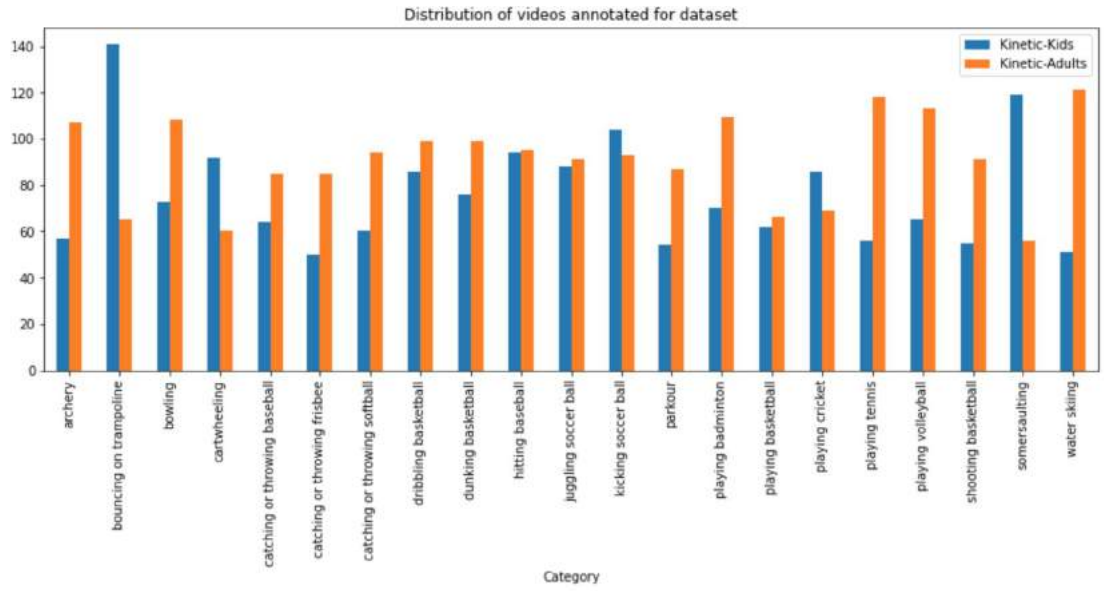

Figure 3.10: Distribution of downloaded videos in Kinetics-Kids and Kinetics-Adults

3.4 Select Clips for Dataset

3.4.1 De-duplication of Clips

We performed the de-duplication step to ensure that we only take one clip from each downloaded YouTube video. After the third annotation, we randomly selected one annotated clip, if we ended up with more than one clip from the same video in each activity class.

While we asked the annotators to indicate if a video contains multiple instances of the activities we are interested in, we only took more than one clip from the video, if the video had the word *"compilation"* in its title. A compilation video usually contains clipped together instances of the same or different activities. We had a total of 15 videos with the title compilation in them.

3.4.2 Train-Test split

After processing all the videos and annotations. We randomly select 40 videos per class to be in the training split, this gives 840 videos for kids training data and 840 videos for adult training data (40 x 21 classes). We then split the remaining videos per category in the ratio of 60 to 40 to be in the test and validation dataset (see Fig 3.11 and Fig 3.12). All of the videos in both datasets are 5 seconds long (See Tab 3.3).

These videos will be used to fine-tune the SOTA models to answer the research questions of this thesis. How this will be done is explained in the next chapter.

	Kinetic-kids	Kinetic-adults
Activity classes	21	21
Total number of videos	1592	1904
Duration per clip	5 seconds	5 seconds
Fps	30	30
Source	YouTube	YouTube
Total duration	133 minutes	159 minutes

Table 3.3: Summary of Kinetic-kids and Kinetic-adults dataset.

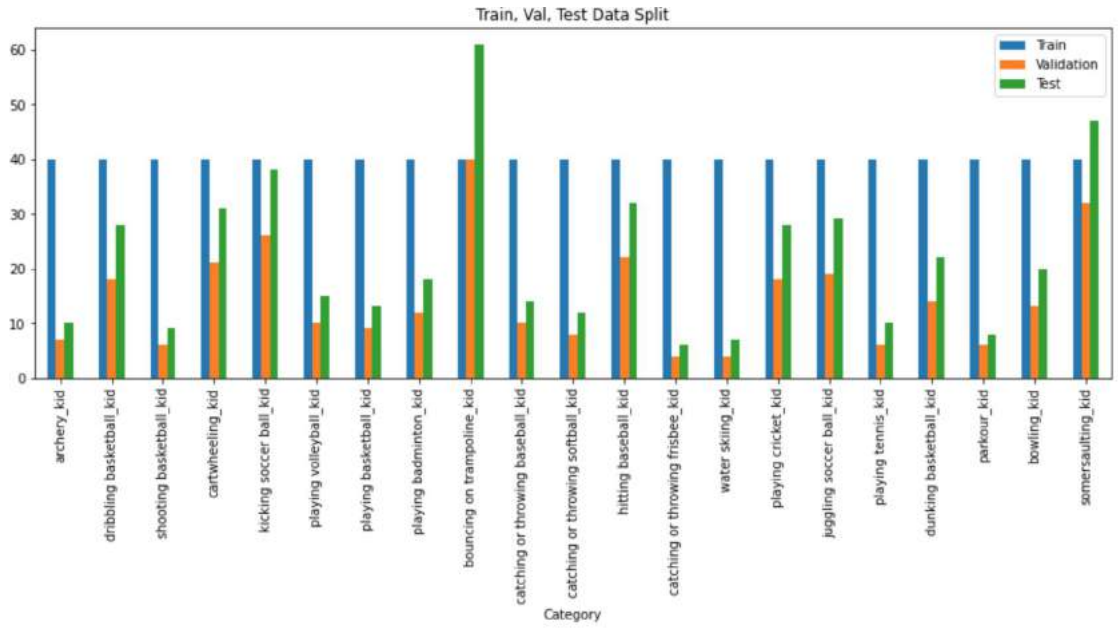

Figure 3.11: Training, Validation and Test split of the Kinetics-kids dataset

3.4.3 Dataset Bias

An important characteristic of datasets for activity recognition is robustness, such that a model trained on one dataset can generalize well on other datasets. Class imbalance is one of the properties that can prevent generalization to other samples. We handle a class imbalance in the Kinetics-kids by ensuring that each class has the same number of videos in their training split (40). However, another source of imbalance that could exist in this dataset is a gender imbalance. Some sports are more prevalent amongst a gender e.g it's more common to find videos of boys juggling soccer than girls. Hence, this might lead to a reduced generalization when the model

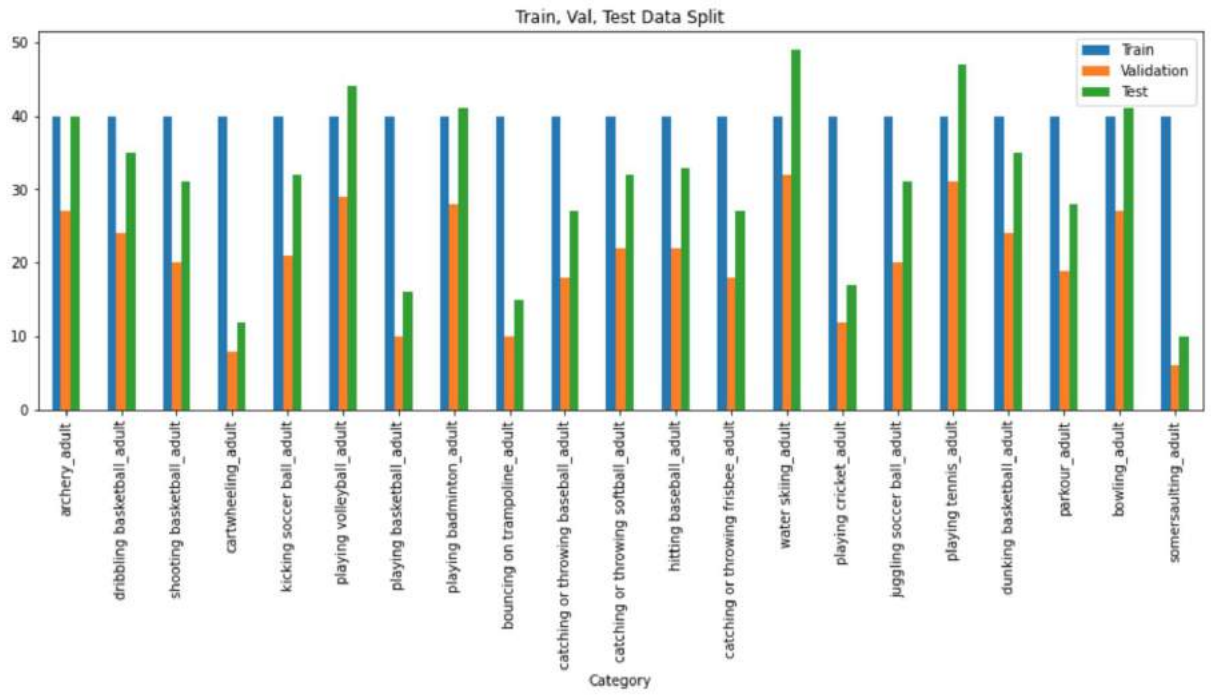

Figure 3.12: Training, Validation and Test split of the Kinetics-Adults dataset

has presented with videos of girls juggling soccer.

Another source of bias in this dataset could be a lack of representation of minority ethnicity. We observed that most of the videos contained kids have western kids and this is because we query YouTube predominantly in the English language. The visual conditions of the video clips could be a source of possible bias. This is because a lot of the videos in the dataset are hand shot and they contain similar motions and lightning conditions. However, applying data augmentation during training could make the model less sensitive to such biases.

Chapter 4

Methodology

This chapter will describe the methods used in answering the research questions (See 1.2). We will be fine-tuning 5 models on the data described in Chapter 3 to answer the research questions. We start by describing our baseline model, Adult Model, the Kid Model, the Mixed model (half-split), and the Mixed model (full-split).

4.1 Baseline Model

A SlowFast model with a ResNet-50 base architecture [Feichtenhofer et al. (2019)] pre-trained on the HACS-clips dataset [Zhao et al. (2019)] serves as the baseline model for all the experiments conducted in this study. The SlowFast model has been used to record state-of-the-art accuracies on activity recognition tasks on benchmark datasets such as AVA v2.1 [Murray et al. (2012)], Charades [Sigurdsson et al. (2018)], and the Kinetics-600 dataset [Carreira et al. (2018)]. This shows that the model is sufficiently complex enough for the experiments we aim to run.

4.1.1 Pre-training Dataset (HACS)

The HACS-clips dataset is a benchmark dataset for activity recognition [Zhao et al. (2019)]. It contains 500k annotated video samples spanned across 200 activity classes. The dataset contains videos downloaded from Youtube. After pre-processing, the dataset contains 1.5M 2-seconds clips. The dataset has been shown to outperform SOTA deep learning model trained on other large-scale datasets like Kinetics-600 [Carreira et al. (2018)] and Sports1M [Karpathy et al. (2014)] dataset when used as a pre-training source [Zhao et al. (2019)]. We choose to use HACS-clips as our pre-training source because of the similarities it has to our dataset such as our video sources being Youtube. Also, the taxonomy used in HACS-clips is derived from ActivityNet [Caba Heilbron et al. (2015)]. While we got our taxonomy from Kinetics-400, the Kinetics dataset shares some taxonomy with ActivityNet. We did not use Kinetics as our pre-training source because we train and test on videos from the Kinetics dataset. Furthermore, we choose HACS-clips over ActivityNet because HACS-clip (500k videos) contains significantly more videos than ActivityNet ($\tilde{2}$0k videos).

While HACS-clips bears some similarities with our datasets, issues such as class imbalance within the HACS-clips dataset could lead to a poor generalization of the learned features to our videos. In Kinetics-kids and Kinetics-adult, we ensured class balance by selecting the same number of training videos per activity class. Another source of bias that could arise with this pre-training source is the lack of representation of minority ethnic groups. We see this as a possible source of bias because the report [Zhao et al. (2019)] did not explicitly state if measures were taken to prevent such bias. Finally, some of the classes present in our dataset are missing from the HACS-clips dataset and this could affect the generalization of the

learned features to our data.

4.1.2 SlowFast Architecture

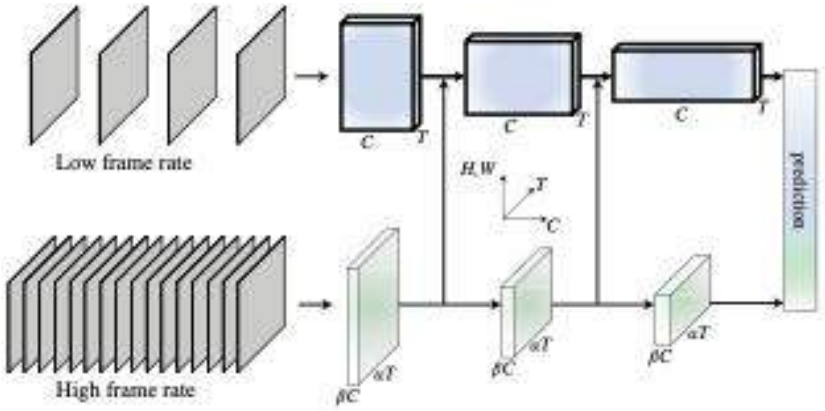

Figure 4.1: SlowFast architecture proposed in [Feichtenhofer et al. (2019)]. Originally shown in [Feichtenhofer et al. (2019)].

The SlowFast model (see Fig 4.1) is a a single-stream model that reads in input at two different frame rates. Unlike two-stream architecture, the SlowFast architecture has a single input source (just frames from the video) but passed through different pathways. This is referred to as the Slow pathway and the Fast pathway. The Slow and Fast pathways can be any convolutional model that works on videos as a spatiotemporal volume, in our baseline, this is a ResNet-50. Both pathways are fused by lateral connection into a SlowFast model [Feichtenhofer et al. (2019)].

The idea behind having two pathways is that the categorical semantics of visual contents do not change as rapidly as with the motion being performed by the subject. For example, the appearance of a subject performing the activity *catching or throwing baseball* doesn't change rapidly over the frames as with the motion of actually throwing or catching the baseball. Other semantic information like the lighting condition or background colors also would not change so rapidly. Hence, the model can learn Spatio-temporal

patterns of shorter and longer duration using both pathways.

The Slow pathway requires low frame rates as input. To do this we take large strides (stride= 16) on our input clip. While the Fast pathway requires a high frame rate, we take a smaller stride (stride=2) for this pathway. Both pathways are kept aware of what each learns by fusing both pathways with a lateral connection after each stage (see Fig 4.1). Our SlowFast implementation was done in PyTorch and can be found in this repository originally written by Alexandros Sterguio (git). Since this study is not about determining which architecture performs best on our dataset, the SlowFast model is sufficiently complex for the experiment we aim to perform.

We will be applying transfer learning to fine-tune the pre-trained model on our dataset. Given the size of our dataset, we will only be fine-tuning the last fully connected layer of the pre-trained model. Since the pre-training source bears some similarities with our data, we expect that the features learned during pre-training should be able to generalize to our dataset. The model contains 34M trainable parameters and we only retrain the last layer of the pre-trained model with 1.5M parameters.

We will start by evaluating the pre-trained model directly on the test split of the Kinetic-kids and Kinetic-adults dataset without fine-tuning. This will allow us to get a sense of how similar the training data used in pre-training is to our data and how useful the learned features from the HACS-clips dataset are for classifying our dataset.

4.2 Adult model

The reason for this study is to see if existing SOTA deep-learning models could generalize to the kids-specific datasets, given that most benchmark activity recognition datasets are adult-specific. We will train an Adult model to test the generalizability of features learned from the adult-specific datasets to the kid-specific datasets. To create the Adult model, we will also fine-tune the last layer of our pre-trained model on only an adult-specific dataset. After the model has been fine-tuned, we will evaluate the adult and kid-specific test split model. This model will be used to answer RQ1 (see 1.2), *"can an adult-specific SOTA model generalize to a kid-specific dataset"*.

4.3 Kid model

We will create a Kid model to answer RQ2 (see 1.2), *"can a kid-specific SOTA model generalize to an adult-specific dataset"*. To do this, we will fine-tune the last layer of our pre-trained network (SlowFast-ResNet50) on the kid-specific dataset described in the last chapter. After training the model, we will first evaluate it on the kid-specific test split. This is to see what classes the model has difficulties with classifying. Next, we will evaluate the Kid model on the adult-specific test split. We will do this to test the generalizability of a Kid model to an adult-specific dataset. This is particularly interesting to see if one of the features the Kid model learns relates to the age of the kid in the video and how the model transfers this when presented with inputs not containing kids.

4.4 Mixed Model (half-split)

As mentioned in Chapter 1, by stating that a lot of benchmark activity recognition datasets are adult-specific, we do not mean that they only contain adult subjects but rather, the majority of subjects performing activities in the dataset are adults. The purpose of creating a Mixed Model is to test if by deliberately ensuring that a dataset has an equal number of kids and adult subjects in it, the model can better generalize to both adult-specific and kid-specific datasets. The Mixed Model (half-split) will be referred to as the MHS model in this study.

The MHS model is created by fine-tuning the last layer of our pre-trained model using a combination of kid and adult-specific datasets. However, we will be using half the training size from each dataset so we end up with the same training size used in both kid and Adult models. That is, for each class in the adult-specific dataset, we only use 20 videos out of the initial 40 videos and likewise for the kid-specific dataset. This will enable us to see how the training size influences the model's predictions. We will evaluate the model individually against the adult-specific test split and the kid-specific test split. This model will be used to answer RQ3 (see 1.2), *"Does training on both kids and adult-specific datasets increase the performance of the model on the adult-specific dataset and kid-specific dataset"*.

4.5 Mixed Model (full-split)

The Mixed model (full-split) is similar to the MHS model. The only difference is that we will train with the full videos available for each class (40 videos) instead of just 20. This means this model is trained with more data

than with the previous models described above. We will create the model by fine-tuning the last layer of our pre-trained model on both the adult-specific and kid-specific training split. The model will be evaluated against the adult-specific and kid-specific test split. The Mixed Model (full-split) model will be referred to as the MFS model in this study.

4.6 Model Implementation Details

The SlowFast-ResNet50 network is the architecture all our models were built upon. We implemented all our models using Pytorch.

We trained the models using Adam optimizer and calculated the loss using the cross-entropy loss function. We set a batch size of 32 and adjust the learning rate (starts at 0.01) using the PyTorch *ReduceLROnPlateau* with a patience value of 80 and a factor of 0.1. The scheduler is called on every batch. Since the scheduler is called on every batch, it reduces the learning rate by a factor of 0.1 whenever there is no improvement in the loss after 80 batches. To prevent over-fitting, we apply early stopping based on the validation loss. If there is no decrease in validation loss after 12 epochs, we stop training the model.

We normalize the shorter image size during training so that it has 384 pixels wide and keeps its aspect ratio on the longer side. With validation and testing, we normalized the shorter image size so that it is 294 pixels wide. We choose these normalizations to match the settings applied to the frames used in pre-training the model. All our videos were resampled to 30fps for consistency in the time between frames when we sample the frames for training and validation. This will ensure that all the segments of the video cover the same amount of time.

We sample 40 frames from the input clip during training and validation. We choose to sample 40 frames because, during pretraining, 16 frames were sampled from 2-sec clips. Since we have 5-sec clips, to match the sampling rate during pre-training we need 40 frames. To make sure the frames sampled spread across a large extent of the clip, we set the stride size as, length of the frame in clips (150) / target number of frames (40). Hence, in this case, we sample every 3rd frame.

We apply the same data augmentation methods as were applied during the pre-training of the baseline model to keep consistency in the inputs we present to the model. During training, we apply uniform cropping of 256x256 to the videos. Other augmentations applied on random include, Gaussian blur, left-right flipping, gamma contrast, linear contrast, hue and saturation, and average blur. During validation and testing, we only apply a center cropping of 256x256 to the video.

Chapter 5

Experimental Results & Discussion

In this chapter, we present the results of our experiment with the 5 models described in Chapter 4 on both of our datasets: adult-specific dataset, and kid-specific dataset. All experiment are evaluated on the test split of the datasets. We will be using the top-1 and top-5 accuracy metrics to evaluate the models.

Top-1 accuracy denotes the accuracy when the actual label of the video is predicted by the model as the most probable classification. **Top-5 accuracy** is the accuracy of when the true label appears in the top-5 most probable options predicted by the model. The top-5 accuracy helps to see the ability of the model in a less strict manner.

We will start by discussing how each model performs on the kid-specific dataset and after that, we present the results on the adult-specific dataset. We will end the chapter by discussing our response to the research questions posed in this study on the basis of the results gotten.

5.1 Kid-specific Test Split

In this section, we discuss how each model performs on this test split. We start by describing how the baseline model performs on the data. Next, we discuss how the Kids model performs, followed by how the Adults model performs. Finally, we discuss how the MHS model and MFS model performs on the kid-specific test split.

5.1.1 Baseline Model Evaluated on Kid-specific Data

We cannot quantitatively evaluate how the baseline model performs on the kid-specific test split because of a mismatch in class labels (200 base labels, 21 target labels). The reason we evaluate this model on the dataset is to get a sense of the kind of features learned from the pre-training source. Also, to see how well the parameters we aim to transfer performs on our dataset.

Overall the features learned on the pre-training source do not generalize well to kid-specific data without fine-tuning (See Appendix). However, there seems to be some pattern as to ehy the model made certain predictions. For example, the model misclassifies the classes *bouncing on trampoline, cartwheeling and parkour* as *doing karate*. While we do not have *doing karate* as one of our acting classes, there are similarities in the motion kids make when doing parkour, cartwheeling, and bouncing on trampoline that bear some resemblance to moves in karate. Hence, even though the baseline could not generalize directly on our sample, it still indicates that the features learned could be useful during the fine-tuning of the other models.

	Kid-specific	**Adult-specific**
Kid Model	43.8%	45.5%
Adult Model	35.0%	51.9%
MHS model	43.8%	51.1%
MFS model	46.2%	52.6%

Table 5.1: Average accuracy reported by all the models on kid-specific and adult-specific dataset

5.1.2 Kid Model Evaluated on the Kid-specific Data

The Kid model records a top-1 accuracy of 43.8% and top-5 accuracy of 78.1% (see Table 5.1). We notice that the top-1 accuracy is particularly low. Given the properties of the dataset such as how varied the actions are in terms of how they are performed and where they are being performed, this makes the dataset somewhat difficult to learn from. The low accuracy suggests that there is a large variation in the test split and that the training split might not containing enough videos to generalize to the variations in the test split.

Aside from the low top-1 accuracy, the model performs well on some classes (see Fig 5.1). One such class is the *water skiing* class. The model achieves a 100% accuracy in this class. While we only have 7 test videos for this class, we can visually observe that that the background of this videos and generally the same (water in sight), also they have similar camera motion. This makes the class a relatively simple one for the model to classify. Furthermore, only two other classes makes a confusion with the *water skiing* class: *somersaulting* and *kicking soccer ball* classes and this misclassification occurs less than 5% of the time in both classes. Visually looking at the misclassified videos, it looked like a random prediction as the background and motion in the video did not look like the ones typically in a *water skiing* video. Another class where the Kid model performs well is the *archery*

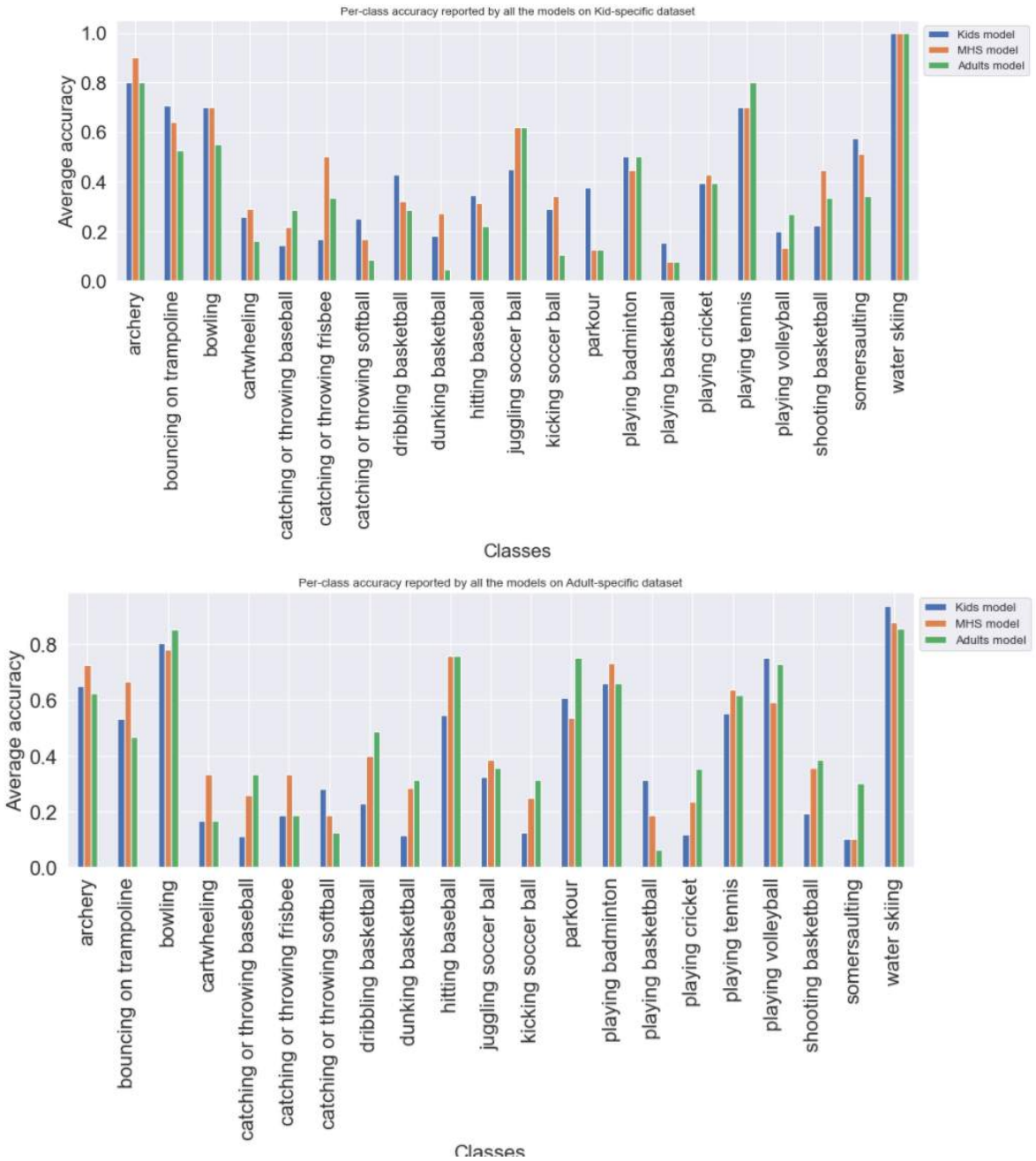

Figure 5.1: Per-class accuracy reported by the Kids model, MHS model, and Adults model on the kid-specific test split (top) and the adult-specific test split (bottom)

class with an accuracy of 80.0%. Like with the *water skiing* class, this class also has relatively little variation in how and where it is performed. Given that *archery* can be a dangerous sport, kids perform this in a supervised environment which is mostly in open fields. From visualizing the model's prediction of this class (see Fig 5.2) we can see that the sport is performed

outside, in a similar pose and a similar background.

Figure 5.2: Archery frames from the kid-specific dataset

The model also classifies the *playing tennis, bowling, and bouncing on trampoline* class with an accuracy of 70.0%. With the *playing tennis* class, the model can predict the standard looking tennis video,i.e., tennis played at the tennis court. However, the model commonly misclassifies the *playing tennis class* with either *catching or throwing a baseball or hitting baseball* (see confusion matrix in Appendix B.3). From the videos misclassified as these, we observe that the environment does look like the same environment where softball and baseball sports are typically performed in this dataset (see Figure 5.9). In general, what we have observed with the classes where the model obtains a high accuracy (above 70.0%) is that the videos in these classes are very similar to each other (color of the court, shape of equipment, similar objects always present in frames and, camera motion) and that the model does have a difficulty in classifying the videos within these classes that look different. We are also aware that having a low number of test sample could influence this.

The three classes that prove the most confusing for the model to classify in the Kid-specific test split are, *catching or throwing baseball* (14.2%), *playing basketball* (15.3%), and *catching or throwing frisbee* (16.6%). With *catching or throwing baseball*, the model commonly confused this with the

Figure 5.3: Playing tennis misclassified as catching or throwing softball by the Kid model

Figure 5.4: Catching or throwing softball frames from the kid-specific dataset

classes *hitting baseball* and *catching or throwing frisbee*. We hypothesized the confusion between *catching or throwing baseball* and *hitting baseball* because both actions usually occurred within a split second of each other in the videos and this also proved difficult to annotators during annotation. However, when we visually inspect the misclassifications made by the model (see Fig 5.6), it misses even scenes where only throwing occurs in the frame. This could be because both action still jointly occur in the training samples. Furthermore, the *catching or throwing baseball* class is one with the classes where the activity is performed in varying environments. This suggests that the model needs more videos to learn from to be able to generalize well to these videos. For the *playing basketball*, the Kid model confuses this class with other basketball classes (*shooting basketball*, *dribbling basketball*

and *dunking basketball*) and the *juggling soccer ball* class. The confusion between *playing basketball* and other basketball classes is understandable due to how similar these classes are to each other. Technically all other basketball classes can be classified as *playing basketball*. In Kay et al. (2017), they also report a confusion between *playing basketball* and *shooting basketball* by a two-stream model trained on Kinetics-400. For the confusion between *playing basketball* and *juggling soccer ball* class, we cannot visually explain why this misclassification could have occurred, because the videos misclassified were *playing basketball*. Visually, we would expect that the presence of the basket is a good indication of playing basketball, however, it is possible that the model finds some similarity in the motion type in the *playing basketball* and *juggling soccer ball* (see Fig 5.5). Finally, with the *catching or throwing frisbee* class, the model seems to find this difficult because it misclassifies the videos with a lot of other classes which makes us think the model did not learn with this class. We also visually compare the videos in the *catching or throwing frisbee* training split to the videos in the test split and they are similar videos (see Fig 5.7). We can not explain the misclassification of the *catching or throwing frisbee* visually.

Figure 5.5: Sample frames from videos in the juggling soccer class (top) and playing basketball class (bottom) in the kid-specific dataset

The top and bottom 5 predictions of the Kid model on the kid-specific test split are presented in Tables 5.2.

In general, the results are in line with our hypothesis that the videos in

Figure 5.6: Catching or throwing baseball videos from kid-specific dataset misclassified by the Kid model

Figure 5.7: Sample frames from videos in the catching or throwing frisbee test split (top) and catching or throwing frisbee train split (bottom) in the kid-specific dataset

the kid-specific dataset are quite complex because of the variations in how and where they are performed. Hence, the model's low accuracy can be attributed to the small training sample and this could be improved with more kid-specific training videos.

5.1.3 Adult model Evaluated on the Kid-specific Data

The Adult model records a top-1 accuracy of 35% and top-5 accuracy of 69% on the kid-specific test split (see Table 5.1). This is a much lower

Class	**Top-1**	**Class**	**Top-1**
Water skiing	100.0%	Catching or throwing baseball	14.2%
Archery	80.0%	Playing basketball	15.3%
Bouncing on trampoline	70.4%	Catching or throwing frisbee	16.6%
Playing tennis	69.9%	Dunking basketball	18.8%
Bowling	69.9%	Shooting basketball	19.9%

Table 5.2: Classes with the highest (left) and lowest (right) top-1 accuracy recorded by the Kid model on the Kid-specific test split

performance when compared to how the Adult model performs on the Adult-specific dataset. One reason for this could be because, the videos in the adult-specific test split are simply different from the videos in the kid-specific test split, which in turn leads to a lower generalization to the kid-specific test split. However, the per-class accuracy obtained by the Adult model on the kid-specific test split shows that the model was able to generalize well to some of the classes in the kid-specific test split (see Fig 5.1).

One such class is the *water skiing* class. The Adult model achieves an accuracy of 100% on this class just like the Kid model. This goes to prove that the *water skiing* class contains relatively simple videos to classify. The Adult model was also able to classify the *archery* and *playing tennis* class with 80% top-1 accuracy. For *archery*, the Kid model also achieves a class accuracy of 80%. However, on the *playing tennis* class, the Kid model only achieves an accuracy of 69.9%. This means the Adult model is slightly better at generalizing to the *playing tennis* class videos in the kid-specific test split than the Kid model. Based on visually analyzing the *playing tennis* videos in the Kid-specific test split (see Fig 5.9) this could be because the number of training samples used in the Kid model was not sufficient for the model to learn all the features needed to generalize to the tennis video in its test split. By visually inspecting the adult-specific *archery* class we observe that the videos contain a more varied background

and lighting conditions than the videos of kids. Also unlike with kids, adults are not constrained to supervision during archery so they perform this in a more varied way than with kids (see Fig 5.8). Furthermore, HACS-clip [Zhao et al. (2019)] which is the pre-training source of the models contains the *archery* and *tennis* classes. Hence we hypothesize that since HACS-clip is also an adult-specific dataset, the Adult model could benefit more from the features learned on HACS-clip as well.

Figure 5.8: Example frames from the Archery class in the Adult-specific dataset

While the Adult model can generalize decently to some of the classes in the kid-specific test split, some classes prove difficult for this model. Some of the classes classified with less than 10.0% by the Adult model are *dunking basketball* (4.5%), *playing basketball* (7.6%) and *catching or throwing softball* (8.3%). Just like with the Kid model, the Adult model also misclassifies the basketball classes with each other. For the *catching or throwing softball* class, the Adult model confuses the class with a lot of other classes which suggests that the model simply could not generalize to this class.

Other classes where the Adult model performs poorly especially since it performs well on these classes in the adult-specific dataset are, *hitting baseball* and *playing volleyball*. On the *hitting baseball* class in the kid-specific test split, the Adult model records an accuracy of only 21.0% while it records an

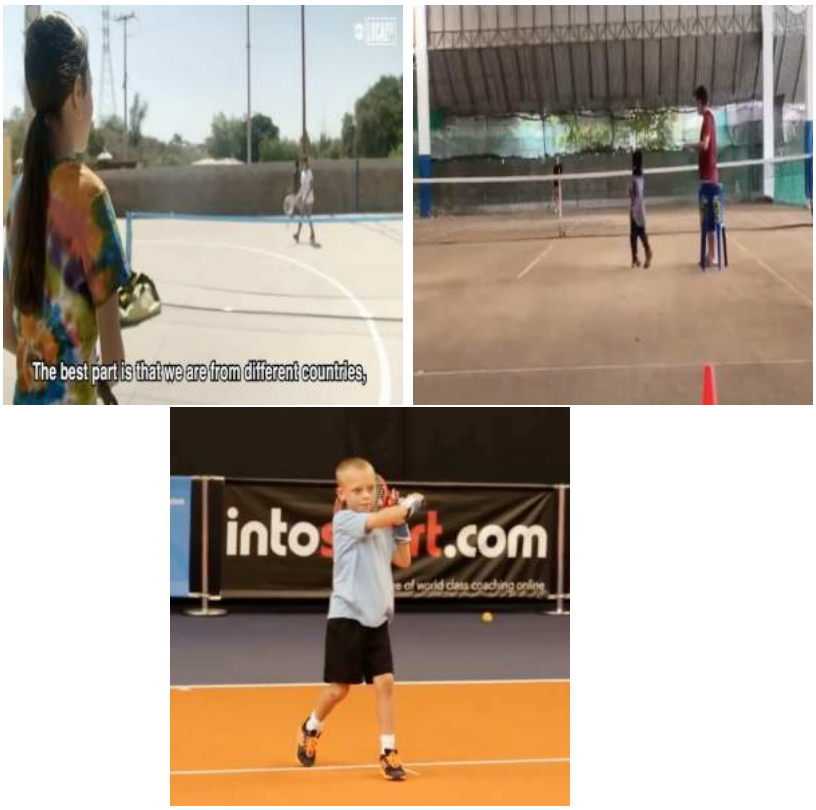

Figure 5.9: Example frames from the Playing tennis class in the Kid-specific test split

accuracy of 75.0% on the same class within the adult-specific dataset. The Adult model commonly misclassifies *hitting baseball* as *catching or throwing softball* within the kid-specific test split. In the misclassified samples (see Fig 5.10), the kids all hold a baseball bat which makes it difficult to say why the model attributes this to *catching or throwing softball*. Although baseballs are supposed to be thrown from an elevated mound while softballs are thrown from a flat designated circle, hence, if the model is unable to decode the baseball bat held by the kid, it can be interpreting the ball and the flat area where the kids are to mean *catching or throwing softball*.

The model also generalizes poorly to the *playing volleyball* class in the kid-specific test split. The Adult model gets a top-1 accuracy of 72.7% on this class in the adult-specific dataset and only 26.6% on the *playing volleyball* class in the kid-specific test split. The model commonly misclassifies *play-*

Figure 5.10: Hitting baseball samples from kid-specific test split misclassified by the Adult model

ing volleyball as *playing basketball* and *shooting basketball* (see examples in Fig 5.11). Like with other classes, the Adult model has a poor generalization to the kid-specific test split, the misclassified samples either occur in a non-standard setting or are played with alternative equipment, e.g., playing volleyball with a yoga-looking ball.

Figure 5.11: Playing volleyball samples from kid-specific test split misclassified by the Adult model

The top and bottom 5 predictions of the Adult model on the kid-specific test split are presented in Tables 5.3.

Class	Top-1
Water skiing	100.0%
Archery	80.0%
Playing tennis	80.0%
Juggling soccer ball	62.0%
Bowling	55.0%

Class	Top-1
Dunking basketball	4.5%
Playing basketball	7.6%
Catching or throwing softball	8.3%
Kicking soccer ball	10.5%
Parkour	12.5%

Table 5.3: Classes with the highest (left) and lowest (right) top-1 accuracy recorded by the Adult model on the Kid-specific test split

5.1.4 MHS model Evaluated on the Kid-specific Data

The model records a top-1 accuracy of 43.8% and top-5 accuracy of 74.6% on the kid-specific dataset. From the accuracy, we can see that, overall, the MHS model generalizes better to the kid-specific dataset than the Adult model does. However, the Kid model and the MHS model record the same top-1 accuracy on the kid-specific dataset (43.8%).

While the MHS model has an equivalent accuracy to the Kid model (see Table 5.1), this model does better than the Adult-specific and Kid model on some of the kid-specific dataset classes (see Fig 5.1). One of these classes is the *archery* class. The MHS model achieves an accuracy of 89.9% in this class while the Kid and Adult model achieves an accuracy of 80.0% in the same class. While we established earlier that the *archery* videos in the kid-specific test split has proven to be a relatively simple class, it can benefit from the features learned from both adult-specific and kid-specific videos to improve generalization to the *archery* class. Especially since we still use the same number of training samples and evaluate on the same test split as was used with the two previous models, the only difference was including both adult and kid-specific videos for training.

Other classes that seem to benefit more from the training data used in MHS model are the *catching or throwing frisbee* and *shooting basketball* class. With the *catching or throwing frisbee*, the MHS model classifies this

class with an accuracy of 50%. This is significantly more than what the Kid model (16.6%) and Adult model (33.3%) achieve in this class. This suggests that by introducing adult-specific *catching or throwing frisbee* samples into the training split, we are better able to generalize more to the frisbee classes in the kid-specific test split. Another class that seems to benefit from the mixed training system is the *shooting basketball* class. The MHS model achieves an accuracy of 44.4% in this class while the Adult model achieves an accuracy of 33.3% and the Kid model achieves an accuracy of 22.2%.

There are also classes within the Kid-specific dataset that do not seem to benefit from including adult-specific videos during training. One such class is the *parkour* class. The MHS model gets an accuracy of 12.5% in this class. This is the same accuracy the Adult model achieves in this class. However, the Kid model achieves an accuracy of 37.5% in the same class. This suggests that the presence of adult-specific *parkour* videos, reduces the generalization of the model to the kid-specific *parkour* class. The kid-specific parkour is one of the classes where we can also easily spot the variation in how and where the activity is being performed, which looks different to what would be considered a standard *parkour* (see Fig 5.12) in the adult-specific test split.

The confusion matrix for this model can be found in Appendix B.5. We present the top-5 classes that had an increase in accuracy when the MHS model is used in Table 5.4 in comparison to the single-training sourced videos.

Figure 5.12: Parkour samples from kid-specific dataset misclassified by the Adult model

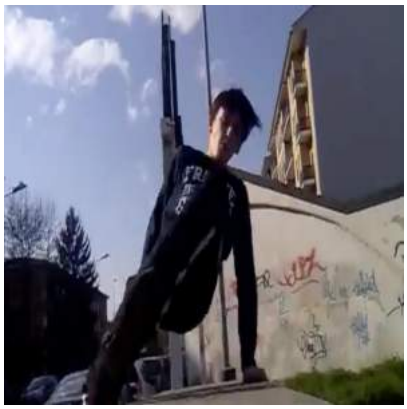

Figure 5.13: Parkour sample from kid-specific dataset correctly classified by the Adult model

Figure 5.14: Parkour samples from adult-specific dataset correctly classified by the Adult model

5.1.5 MFS model Evaluated on the Kid-specific Data

The MFS model records a top-1 accuracy of 46.2% and top-5 accuracy of 78.0% on this model. Overall, these are the best classification results

Class	**MHS model**	**Kid model**	**Adult model**
Catching or throwing frisbee	50.0%	16.6%	33.3%
Shooting basketball	44.4%	22.2%	33.3%
Playing cricket	42.8%	32.9%	32.9%
Kicking soccer ball	34.2%	28.9%	10.5%
Dunking basketball	27.2%	18.1%	4.9%

Table 5.4: The top 5 classes that improve in accuracy when MHS model is used on the kid-specific test split

recorded on the Kid-specific dataset. While the MHS model was only able to match the performance of the Kid model on the top-1 accuracy (see Table 5.1), the MFS model surpasses the Kid model on both top-1 and top-5 accuracy. Based on these figures, there is evidence that the Kid model benefits from having a mixed training dataset and increasing the number of training videos.

Some of the classes where the MFS model performs better than the MHS model includes the *bouncing on trampoline* and *catching or throwing baseball* class. On the *bouncing on trampoline* class, the MFS model achieves an accuracy of 75.4% while the MHS model achieves an accuracy of 63.9%. While this is a substantial improvement from the MHS model, we think this improvement is because there was an increase in the number of kid-specific trampoline videos and not necessarily because of the presence of more adult-specific trampoline videos. The reason why we make this argument is that the Kid model was already able to achieve an accuracy of 70% on the *bouncing on trampoline* class, hence, by increasing the training samples of the Kid model we could get an increase in the performance of the Kid model to this class. Another class where the MFS model does better than other models is the *catching or throwing baseball*. The model records an accuracy of 35.7% on this class while the MHS model got an accuracy of 21.4% on this class.

Like with MHS model, there are classes where the MFS model records a low accuracy. One of such classes is the *bowling* class. The MFS model records an accuracy of 60.0% on this class while the MHS model got an accuracy of 69.9% on the same class. By comparing the performance of the Kid model (69.9%) and the Adult model (55.0%) to both versions of the Mixed model (MHS, MFS), we cannot say for certain that the bowling class benefited from the model having a mixed data for training. Another class where we see a decline in accuracy when classified with the MFS model is the *catching or throwing frisbee* class. The model records an accuracy of 33.0% on this class while the MHS model got an accuracy of 50.0%.

The confusion matrix of the performance of this model on the kid-specific dataset can be found in Appendix B.8. We present the classes that had an increase in accuracy when the MHS model is used in Table 5.5.

Class	**MFS model**	**MHS model**	**Kid model**	**Adult model**
Bouncing on trampoline	75.4%	63.9%	70.4%	52.4
Playing badminton	50.0%	44.4%	49.9%	49.9%
Catching or throwing baseball	35.7%	21.4%	14.2%	28.5%
cartwheeling	35.4%	29.0%	25.8%	16.12%

Table 5.5: The classes that improve in accuracy when MFS model is used on the kid-specific test split

5.1.6 Summary of the model's performance on the Kid-specific dataset

We ran a total of 5 models on the Kid model. We ran the baseline to see how useful the features learned from the pre-training source could be to this dataset. Next, we ran the Kid model, Adult model, MFS model, and MHS model to determine which type of training data generalizes better

to the kid-specific dataset. From the top-1 accuracies reported, the MFS model generalizes best to the kid-specific dataset (see Table 5.6).

Model	**top-1**	**top-5**
Kid model	43.8%	78.1%
Adult model	35.0%	69.0%
MHS model	43.8%	74.6%
MFS model	46.2%	78.6%

Table 5.6: Top-1 and top-5 accuracy of all the models evaluated on the kid-specific test split

The results discussed above points to the complexity of data present in the kid-specific dataset. Given the complexity of the dataset and the fact that we only fine-tune the last layer of the baseline for each model, a larger training sample and training more layers could be a way to improve the performance of the SOTA deep-learning recognition model on the kid-specific dataset.

In the next section, we will discuss the results of the models when evaluated against the adult-specific dataset.

5.2 Adult-specific Test split

In this section, we discuss how each model performs on this test split. We start by describing how the baseline model performs on the data. Next, we discuss how the Adult model performs, followed by how the Kid model performs. Finally, we discuss how the MHS model and MFS model performs on the Adult-specific test split.

5.2.1 Baseline Model Evaluated on Adult-specific Data

Even though we consider the pre-training source (HACS-clips) an Adult-specific dataset, the model does badly at generalizing what it has learned to the Adult-specific dataset without fine-tuning. We choose this dataset as our training set because of the similarities it shared with Kinetics. However, since we only use a subset of Kinetics-400 in which a large subset of it does not exist in HACS-clips, this was not the most suitable choice to transfer learn on.

Nonetheless, the features still prove useful even though we only fine-tune the last layer. With a larger training split, fine-tuning more layers from this model could help improve the overall accuracy of all our experiments.

5.2.2 Adult model Evaluated on the Adult-specific Data

The Adult model records a top-1 accuracy of 51.9% and top-5 accuracy of 81.9%. While this would not be considered a high accuracy, the performance of the Adult model on the Adult-specific dataset is generally higher than its performance on the Kid-specific dataset (35.1%). Also, the size of the training sample could be a contributing factor to as to why the model performs low. Based on the model's better performance on the Adult-specific test split, there is an indication that the Adult-specific test split is less complex than the Kid-specific test split. Another factor that could have contributed to an increase in this performance is the pre-training source. Since the pre-training source is also an Adult-specific dataset, the features learned from this will better generalize to other adult-specific data than to kid-specific data.

While the model does not predict any of the classes 100% correct, the model does present high accuracy for some of the classes (see Appendix B.1). For example, the model was able to predict the class *bowling* with 85.37% and, *water skiing* with 85.71%. We expect that these classes are easier to predict because they have less variation in the environment in which they are performed or how they are performed. An adult would always bowl in a bowling alley and water skiing is always performed on water. Also, the Adult model did significantly better at classifying the *bowling* class in the Adult-specific test split, than it did on the *bowling* class (55.0%) in the kid-specific test split (see Fig 5.1). This is understandable as there is generally more variation in how and where kids bowl than with adults, so visually the videos look different (see Fig 5.15).

Figure 5.15: Bowling samples from the Kid-specific dataset

Figure 5.16: Bowling samples from the Adult-specific dataset

Other classes where the Adult model performs significantly better on the

classes within the Adult-specific test split than it does on the class in the kid-specific test split are *playing volleyball* and, *playing badminton*. The Adult model records an accuracy of 72.0% on the *playing volleyball* class in the adult-specific test split, and only achieves an accuracy of 26.0% on the *playing volleyball* class in the kid-specific test split. Visually inspecting the videos in the *playing volleyball* class in the kid-specific test split (see Fig 5.11), we see that these videos do not look like the standard volleyball videos in the Adult-specific dataset.

However, there are also classes within the Adult-specific dataset that prove difficult for the Adult model to classify. One of such classes is the *catching or throwing softball* (12.5%) class in the adult-specific test split. The model confuses the *catching or throwing softball* with *catching or throwing baseball* (15.62%) and *kicking soccer ball* (12.5%) (see examples in Fig 5.17). While *catching or throwing softball* and *catching or throwing baseball* share visual similarities in how they are played, the model seems to be encoding green grass as one of the features for *kicking soccer ball*. This is because the samples that were misclassified as *kicking soccer ball* happen in an environment similar to a soccer field (see Fig 5.18).

Some of the classes where the Adult model does better on the Kid-specific test split than it does on the Adult-specific test split are *archery* and *juggling soccer ball* class. As previously mentioned, the Adult model can generalize better to the *archery* class in the kid-specific test split (80.0%) than it does to the *archery* in the adult-specific test split (62.5%). This is one of the classes in the Adult-specific dataset that contains a lot of variation in how archery is performed, as opposed to other standard sports (see Fig 5.8). The model also performs better on the *juggling soccer* class in the kid-specific test split than it does on the *juggling soccer* class in the adult-specific test split. The model achieves an accuracy of 35.4% on the

Figure 5.17: Catching or throwing softball clips misclassified as catching or throwing baseball by the Adult model

Figure 5.18: Catching or throwing softball clips misclassified as kicking soccer ball by the Adult model

class in the adult-specific test split and accuracy of 62.0% on the *juggling soccer* kid-specific test split. However, we observe that this not a case of having so many variations in the dataset, we think the model just needs more training samples to better learn this class.

The top and bottom 5 predictions of the Adult model on the adult-specific test split are presented in Tables 5.7.

Generally, the Adult model does better on the classes within the Adult-specific test split than on the same classes in the Kid-specific test split. This falls in line with our hypothesis, that this happens because the Adult-specific dataset is less complex than the Kid-specific dataset. While we have generally argued that the complexity of videos in the dataset is a

Class	Top-1
Water skiing	85.7%
Bowling	85.3%
Hitting baseball	75.7%
Playing volleyball	72.7%
Playing badminton	65.8%

Class	Top-1
Playing basketball	6.2%
Catching or throwing softball	12.5%
Cartwheeling	16.6%
Catching or throwing frisbee	18.5%
Somersaulting	30.0%

Table 5.7: Classes with the highest (left) and lowest (right) top-1 accuracy recorded by the Adult model on the Adult-specific test split

major reason why the Adult model can generalize better to the Adult-specific dataset than it does on the Kid-specific dataset, the training size could also be a factor. With the more adult-specific video for training, there is a possibility that the model can learn features that will generalize to the Kid-specific dataset.

5.2.3 Kid model Evaluated on the Adult-specific Data

The Kid model records a top-1 accuracy of 45.5% and a top-5 of 76.2% on the Adult-specific dataset. We see that the Adult model performs better than the Kid model on the Adult-specific dataset. However, the Kid model was able to better generalize to some of the classes than the Adult model.

One such class is the *catching or throwing softball* class in the adult-specific test split. The Kid model classifies this class with an accuracy of 28.0%. While this is low accuracy, the Adult model does even lower by only being able to classify the class with a 12.5% accuracy. What we observe is that the model commonly misclassifies this as *hitting baseball*, which is the confusion the Kid model also makes on the Kid-specific test split. We think that the reason why the Kid model can generalize more to the Adult-specific test split is that most of the videos in the *catching or throwing softball* class do not have the hitting activity occurring in the same frame as with the *catching or throwing baseball* in the kid-specific test split. Other classes

Class	Top-1
Water skiing	93.8%
Bowling	80.0%
Playing volleyball	74.9%
Playing badminton	65.8%
Bowling	64.9%

Class	Top-1
Somersaulting	10.0%
catching or throwing baseball	11.1%
Dunking basketball	11.4%
Playing cricket	11.7%
Kicking soccer ball	12.5%

Table 5.8: Classes with the highest (left) and lowest (right) top-1 accuracy recorded by the Kid model on the Adult-specific test split

where the Kid model performs better than the Adult models includes, *bouncing on trampoline* and *playing basketball*. These results suggest that the presence of variations in the kid-specific training set makes the Kid model generalize better to the Adult-specific test split.

While the Kid model can generalize well to some of the classes, it still does badly on some of the others. Some of these classes includes, *somersaulting* (10.0%), *catching or throwing baseball* (11.1%) and *dunking basketball* (11.4%). The Kid model commonly misclassifies the *somersaulting* class as *cartwheeling* within the adult-specific test split. Even though both classes are visually similar, the Kid model can distinguish between the *somersaulting* and *cartwheeling* class in the kid-specific test split. By visually examining the *somersaulting* in the adult-specific test split, the activity is mostly carried out at a gymnastics gym, while the *somersaulting* videos in the kid-specific test split occur in a home setting. We think this makes it difficult for the Kid model to classify the *somersaulting* class in the adult-specific test split because it does not associate the gymnastic gym with the activity *somersaulting*.

We present the top and bottom 5 predictions of the Kid model on the adult-specific test split are presented in Tables 5.2. The model's confusion matrix on this split can also be found in Appendix B.4.

5.2.4 MHS model Evaluated on the Adult-specific Data

The MHS model achieves a top-1 accuracy of 51.1% and top-5 of 81.1% on the Adult-specific dataset. This performance is slightly lower than that of the Adult model (51.9%). Even though the MHS model generally performed a little lower than the Adult model on the Adult-specific test split, there are classes where the model performs better at classification than the Kid and Adult model.

One of such classes is the *archery* class. The MHS model achieves an accuracy of 72.5% in this class while the Kid and Adult models both achieve an accuracy of less than 65.0% in the same class. We mentioned above that the archery class within the Adult model seems to contain more complex sample videos than those in the Kid-specific test split. This can also be proved by the fact that all the models (Kid, Adult, MHS and MFS models) were able to classify the archery class within the kid-specific test split with an accuracy of over 80.0%. Hence, the ability of the MHS model to classify the *archery* class within the adult-specific test split with an increased accuracy suggests that the *archery* class does benefit from including kid-specific data in the training sample. Another class the MHS model model performs better at in the adult-specific test split is the *playing badminton* class. The model achieves an accuracy of 73.1% in this class while the Adult and Kid models both achieve an accuracy of 65.8%. Also, the MHS model performs less (44.4%) when classifying the *playing badminton* class in the kid-specific test split, which suggests that including kid-specific data helps the model better classify the *playing badminton* class in the adult-specific test split. Other classes that seem to benefit from having both kid and adult-specific training samples are *bouncing on trampoline* and *cartwheeling*. We think

these classes in the Kid-specific dataset introduces more complexity to the training samples used in the MHS model because they have more variations than the class in the adult-specific test split. Also, these variation seems to help in improving the model's generalization to the classes.

There are also classes within the Adult-specific dataset that get a reduced classification accuracy with the MHS model. The model achieves an accuracy of 53.5% on the *parkour* class in the while the Adult model was able to achieve an accuracy of 75.0% on the same class and the Kid model achieves an accuracy of 65.8%. The drop inaccuracy on the *parkour* class when the MHS model and Kid model is used, suggests that adding kid-specific data to training is not beneficial to classifying *parkour* class in the Adult-specific test split. This is interesting because the accuracies of the models (Kid model, Adult model, and MHS model) on the *parkour* class in the kid-specific test split. It also suggests that including parkour data from the Adult-specific dataset in training is non-beneficial for classifying the *parkour* class in the kid-specific test split. Based on Fig 5.12, we think this is because of the difference in the environment and possibly how parkour is done by adults as opposed to kids. Another class, the MHS model classifies with lower accuracy than the other models is the *playing volleyball* class in the adult-specific test split. The model records an accuracy of 59.0% while the Adult and Kid models both record an accuracy greater than 70.0%.

The confusion matrix of the performance of this model on the Adult-specific test split can be found in Appendix B.6. We present the top-5 classes that had an increase in accuracy when the MHS model is used in Table 5.9.

Class	MHS model	Kid model	Adult model
Playing badminton	73.17%	65.8%	65.8%
Archery	72.5%	64.9%	62.5%
Bouncing on trampoline	66.6%	53.3%	46.6%
Juggling soccer ball	38.7%	32.2%	35.4%
Cartwheeling	33.3%	16.6%	16.6%

Table 5.9: The top 5 classes that improve in accuracy when MHS model is used on the Adult-specific test split

5.2.5 MFS model Evaluated on the Adult-specific Data

The model records a top-1 accuracy of 52.6% and top-5 of 85.2% on the Adult-specific dataset. This is the highest top-1 and top-5 accuracy recorded across all the models in our experiment (see Table 5.1). However, this is only a slight increase in accuracy from MHS model's performance on the Adult-specific test split. This does not show substantial evidence that the Mixed model's (MHS and MFS) generalization to the Adult-specific dataset benefits from increasing the training size. Especially since the size of the training videos was doubled and we only get a top-1 increase of less than 1.0%. Nonetheless, there are classes within the Adult-specific test split that indicate they benefit from increasing the training sample in the MFS model.

One such class is the *playing volleyball* in the adult-specific with an accuracy of 79.0%. In comparison to the MHS model which got an accuracy of 59.0%, increasing the training proves to be beneficial to this model. Furthermore, the *playing volleyball* class in the adult-specific test split seems to benefit from both features learned on the kid-specific and adult-specific dataset. We say this because the Kid and Adult models both record accuracy of over 70.0% on this class. The second class is the *dribbling basketball* class. The MFS model got an accuracy of 51.4% on this class while the MHS model got an accuracy of 40.0%. It is again difficult to particularly

say if the increase in accuracy is as a result of an increase in training samples for both the kid and adult-specific sample or just an increase in the adult-specific sample. This is because the Adult model was already able to achieve an accuracy of 48.0% while the Kid model only achieves an accuracy of 22.8% in this class. The last class is the *cartwheeling* class in the adult-specific test split. The MFS model records an accuracy of 41.6%. The model is also the best model at classifying the *cartwheeling* class in the kid-specific test split.

For the classes that experience a drop in accuracy when the MFS model is used, this seems to happen because of the presence of one class over the other. So some classes seem to benefit more from the Adult-specific dataset than the Kid-specific dataset and vice-versa. As mentioned earlier, we can only know for sure if an extra experiment is run on the mixed models to determine the influence of each data-type (adult-specific and kid-specific dataset) to the generalization capability of the model.

The confusion matrix of the performance of this model on the Kid-specific dataset can be found in Appendix B.7. We present the classes that had an increase in accuracy when the MHS model is used in Table 5.10.

Class	**MFS model**	**MHS model**	**Kid model**	**Adult model**
Playing volleyball	79.5%	59.0%	74.9%	72.7%
Dribbling basketball	51.4%	40.0%	22.8%	48.5%
cartwheeling	41.6%	33.3%	18.5%	18.5%

Table 5.10: The classes that improve in accuracy when MHS model is used on the Adult-specific test split

5.2.6 Summary of the model's performance on the Adult-specific dataset

Like with experiments conducted on the Kid-specific dataset, we also run 5 models on the Adult-specific dataset. The baseline model, Adult model, Kid model, MHS model, and MFS model. From the top-1 accuracies reported, the Mixed-model (full-split) generalizes best to the Adult-specific dataset (see Table 5.11).

Model	**top-1**	**top-5**
Adult-specific	51.9%	81.9%
Kid-specific	45.5%	76.2%
MHS model	51.1%	81.1%
MFS model	52.6%	85.2%

Table 5.11: Top-1 and top-5 accuracy of all the models evaluated on the Adult-specific dataset

Generally, the results indicate that the videos in the Adult-specific dataset are easier to classify in comparison to the Kid-specific dataset. Also, while the Mixed-model (full-split) records the best accuracy on the Adult-specific dataset, we hypothesize that the model might be able to do better than the mixed model in the currents form with more training data.

In the last section, we give a summary of the results reported on both datasets with respect to the research question posed at the beginning of this research.

5.3 Research Questions

In this chapter, we provide answers to our research questions based on the results presented above.

We will answer the research question chronologically from RQ1 - RQ3 and then we discuss the main research question posed by this study.

5.3.1 RQ1: Can an Adult SOTA model generalize to a Kid-specific dataset?

Based on the accuracy recorded (see Table 5.1), we see a drop in the performance of the Adult model when it is evaluated on the Kid-specific test split as opposed to when it is evaluated on the Adult-specific test split.

While the Adult model has proven to be able to generalize to some classes within the Kid-specific test split, the overall performance of the Adult model on the kid-specific test split (35.15%) is low. One of the factors that could explain this is that the kid-specific test split generally contains more variations in how and where each sport is performed than the adult-specific test split. Another possible reason is that the adult-specific data were all downloaded from the Kinetics-dataset and a large part of the data has a relatively low resolution (down to 144p) while the videos we download for the kid-specific data generally have a higher resolution. Hence, since the Adult model was trained on generally lower resolution images, it might find it difficult to generalize to higher resolution videos in the kid-specific test split. However what seems the most plausible factor is that the Adult model could not generalize to the kid-specific test split because the videos in the adult's training sample look different from those in the kid-specific test split.

Based on the accuracy alone, we can say that there is a decline in the performance of an Adult model and the generalization to a kid-specific test split is low. Given the size of the training samples we have in this dataset,

we would expect some variation in the performance of the model, if trained multiple times. However, we observed that the model is quite stable at its performance as it was returned accuracy within the same range even when trained multiple times. This could suggest that the model needs more data to be able to grow in performance. However, further experiments will be required to determine which factors influence the model's ability to not generalize currently to the Kid-specific test split.

5.3.2 RQ2: Can a Kid-specific SOTA model generalize to an Adult-specific dataset?

The Kid model generalizes better to the Adult-specific dataset than the Adult model does to the Kid-specific dataset. As we already hypothesized that the Kid-specific dataset generally contains more variation in terms of where and how the activity is performed, hence, the model could have learned complex and general features that help it to generalize well to the Adult-specific dataset. Another possible factor could be the resolution of the videos used in training our Kid model. A large part of the videos has a resolution of up to 720p while a large number of the videos in the Adult-specific dataset have a lower resolution than this, which could be why the Adult-specific dataset generalizes poorly to kid-specific data.

In general, we believe that even though the Kid model recorded a low accuracy on the Kid-specific test split, it is complex enough to generalize on the Adult-specific test split. As explained in the results chapter, the reason why we think the Kids model performs lowly on the Kid-specific dataset is because the number of training videos available is not enough for the video to learn how to generalize to the test split.

5.3.3 RQ3.a: Does training on both kids and adults-specific datasets increase the performance of the MHS model on Adult-specific Data?

The accuracies reported (see Table 5.1) show that the MHS model performs a little lower than the Adult model on the Adult-specific test split which implies that using both data types does not improve the performance of the model on the Adult-specific test split. However, the MHS model achieves this accuracy only using half the size of adult-specific training data used by the Adult model.

There are indications that certain classes can benefit from having both kid-specific and adult-specific video in the training sample but there are also classes that have reduced accuracy possibly because both data source (kid and adult-specific videos) was used during training. This makes it difficult to make a general conclusion as to whether using both adult and kid-specific data in training is beneficial for generalizing to the Adult-specific dataset.

Like we said with the use of the MHS model on the Kid-specific dataset, additional experiments should be conducted to find out what balance of kid and Adult-specific training data is needed for a better generalization to Adult-specific datasets and also if including the Kid-specific data is at all needed for increased generalization to Adult-specific datasets.

5.3.4 RQ3.b: Does training on both kids and adults-specific datasets increase the performance of the MHS model on Kid-specific Data?

Based on the accuracy alone, we see evidence that the MHS model generalizes as well as the Kid model to the Kid-specific dataset. The MHS model even records a better accuracy than both the Adult and Kid models in some of the classes. Another indicator that training on both kid-specific and adult-specific data is beneficial to generalizing to kid-specific data is that the MHS model only contains half the amount of kid-specific data that was used in training the original Kid model and it was still able to match the performance of the Kid model. Furthermore, even though the adult-specific dataset seems relatively simple in comparison to the kid-specific counterpart, there was evidence above that suggests some of the classes in the adult-specific class contains more variation that could have played a part in the MHS model ability to generalize to some of the classes in the Kid-specific dataset. However, we also see classes where the MHS model recorded lower accuracies. This suggests that additional experiments should be conducted to find out what balance of kid and adult-specific training data is needed for a better generalization to Kid-specific datasets. That is, do we need more kid-specific training data and less adult-specific training data for this model or vice-versa. Such an experiment could also indicate that having an equally balanced kid and adult-specific training sample is the best training option for the model.

5.3.5 RQ3.c: Does increasing the size of training data in the mixed model (MHS) improve generalization to the Kid-specific dataset?

Overall, there is an increase in the classification of the Kid-specific test split when the MFS model is used. This indicates that by increasing the number of kid-specific and adult-specific training videos, the model can generalize better to the Kid-specific test split.

However, just like with the MFS model, more experiments have to be performed to verify the right balance between the two types of data that provides the best generalization to the Kid-specific test split.

5.3.6 RQ3.d: Does increasing the size of training data in the mixed model (MHS) improve generalization to the Adult-specific dataset?

In general, the adult-specific dataset does not seem to benefit from having both kid-specific and adult-specific data in the training sample, at least in comparison to the increase in accuracy we saw when the Mixed Model (full-split) is evaluated on the Kid-specific dataset. We will say the Adult model seems to do best on the adult-specific data. The difference in the level of variation in the adult-specific dataset versus the Kid-specific dataset is not the only factor contributing to the generalization of the model. Other factors could be, the resolution of videos in the dataset, typical camera motion in the videos, backgrounds, etc.

5.3.7 MRQ: Do the current state of the art (SOTA), deep learning models, for activity recognition generalize to kids-specific dataset?

Looking at the accuracies recorded by all the models used in this study, especially the Kid Model, we can conclude that the current STOA deep learning model for activity recognition is sufficient to learn from and generalize to a Kid-specific dataset. However, a larger kid-specific dataset will be needed to see how much the current SOTA deep learning models can learn from a Kid-specific dataset and if they can record benchmark results on a Kid-specific dataset.

Chapter 6

Limitations & Future Work

The experiment performed in this study has shed more light on the biases and possible usefulness of kid-specific datasets for performing activity recognition. However, there are some areas of our work that can be improved. Below we discuss these areas and give possible solutions to them.

6.0.1 Selecting Activity Classes

One of the limitations of our result is that we only consider only sporting activities. We only use sports classes because it is one activity group where we can easily spot the differences between kids and adults. However, if we are to create activity recognition applications that recognize kids doing an activity, more classes would have to be considered in such a study as this. Furthermore, the model had a hard time classifying hierarchical labels like *playing basketball, shooting basketball, dribbling basketball, and dunking basketball*. For further studies, it is sufficient to consider all these classes as just *playing basketball*. In essence, when compiling activity labels for activity recognition, having hierarchical labels should only be considered if

there is a clear difference between the activities.

6.0.2 Compiling Queries Youtube

One limitation of the videos we have downloaded for the kid-specific dataset is the lack of cultural diversity. While the Youtube videos are mostly reflective of the real world, it is quite biased towards western contents. This means that the majority of the videos in the kid-specific dataset contains caucasian children, more specifically American children. This begs the question, would this kids-model be able to generalize to kids from other races performing the same activity. During the future collection of kid-specific datasets, we suggest translating the queries into other common languages used on Youtube. Also, multiple download sources should be considered e.g. Youku, Daum tvPot, etc. This would help to increase the diversity in the dataset and ensure that a racially biased model is not created.

6.0.3 Downloading more Videos

The results of this study were limited by the number of training samples we had. One of the downsides of using a deep learning model is that they require a large amount of training sample even when a pre-trained model is used. We expected that there would be a lot of variation in performance when training a model several times due to our small-sized training samples. However, the training and validation accuracy always fell within the same range when they were trained multiple times. This suggests that the model does need more data even when we only fine-tune the last layer. For the further collection of videos for a Kid-specific dataset, other sources like

social media sites can be considered as a source of download since they typically contain a lot of postings about kids and they are scrapped by using hashtags. Another source of download could be news websites and kids' advertisement videos. Another possible data source could be animation videos (see Covre et al. (2019)). While more studies would be required before this data source can be used, animation videos are modeled after kids and might provide motion information needed by the model to classify kids' activity.

6.0.4 Pre-processing Step: People Detector and Bounding Boxes

One of the pre-processing steps that could have introduced bias into the dataset was using a People detector (pre-trained YOLO-V3 [Redmon and Farhadi (2018)]). We apply this filtering process to limit the number of noisy videos presented to the video annotators. However, the people detection model used was also pre-trained on the adult model, hence, we could have discarded a lot of videos because the detector could simply not recognize children in the scene. Also, it is possible the threshold set on the bounding boxes does not account for children because they had smaller pixels than the threshold set on the bounding box.

6.0.5 De-dupliaction of Dataset

While we de-duplicate videos by ensuring that we only download one video per youtube link, we did not do this across classes. One way we could have addressed this was to build a feature vector for sampled frames from each video and then find the cosine similarities between the videos. The

threshold of similarity can be tweaked according to the activity class. This worked for de-duplication in the Kinetics-400 [Kay et al. (2017)].

6.0.6 Temporal Extent of Activity

Also, within the data collection pipeline, we encountered difficulty with determining what constitutes the start of an action. This means we got different start times annotated for the same video. While we tried to correct this by having an additional annotator and taking average time, it is still worth studying how to better define the temporal extent of activities. Especially since there seems to be a difference in the duration of each activity.

6.0.7 Video Resolution

One of the factors that could affect the generalization of each model to the opposing dataset (Kids model-Adult dataset & Adults model-Kid dataset) is the resolution of the videos in both datasets. since we have varying resolutions across classes and across the datasets, further ablation studies would be required to determine how much effect the resolutions of videos in the datasets affect the generalization ability of the model.

6.0.8 Pre-trainig Source

Furthermore, given that we had a small training size, we could have considered choosing a more similar pre-training source to transfer-learn on. While there is no benchmark kid-specific dataset yet, we could have made sure that the majority of our classes also existed in the pre-training source as

this could have given us better results. Another thing we could have done was to download our pre-training data ourselves. This is because using a pre-trained network containing more classes than we have, could also be a source of confusion for the model at classifying our datasets. Having a pre-training source contain the same sports classes that we had would have made for a better baseline model.

6.0.9 Further Abalation Studies

In general, a comprehensive ablation study is needed to understand the properties of the kid-specific dataset, such as the temporal extent of activities, video resolution, training size, and semantic information, and motion information. By understanding how these properties of the Kid-specific dataset differ from that of the Adult-specific dataset, it becomes easier to understand the model's behavior on these datasets and see which features influence each model the most. Furthermore, an ablation study to test the influence of the different factors associated with each dataset could help come up with a training scheme that helps both adult-specific and kid-specific datasets benefit more from each other's complexity.

Chapter 7

Conclusion

To conclude, this thesis investigated if SOTA deep learning models for action recognition can generalize to the kid-specific dataset. To do this we created a kid-specific and an adult-specific dataset. We also present our data collection pipeline to foster the future collection of kids-specific data.

Our results show that, while SOTA deep learning can be used to classify kid activities, the kid-specific dataset is more complex to generalize to than the adult-specific dataset. The study also shows that the features learned from training on a kid-specific dataset alone can be used to classify adult activities while the reverse is not the case. More work is needed to determine what properties make the kid-specific model generalize better to adult-specific datasets.

Bibliography

Aggarwal, J. K. and Ryoo, M. S. (2011). Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 43(3):1–43.

Aggarwal, J. K. and Xia, L. (2014). Human activity recognition from 3d data: A review. *Pattern Recognition Letters*, 48:70–80.

Ahmadi, M., O'Neil, M., Fragala-Pinkham, M., Lennon, N., and Trost, S. (2018). Machine learning algorithms for activity recognition in ambulant children and adolescents with cerebral palsy. *Journal of neuroengineering and rehabilitation*, 15(1):105.

Akila, K. and Chitrakala, S. (2018). Managing interclass variation in human action recognition. In *International Conference for Phoenixes on Emerging Current Trends in Engineering and Management (PECTEAM 2018)*. Atlantis Press.

Ali, S. and Shah, M. (2008). Human action recognition in videos using kinematic features and multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, 32(2):288–303.

Argyriou, A., Micchelli, C. A., Pontil, M., and Ying, Y. (2007). A spectral regularization framework for multi-task structure learning. In *NIPS*, volume 1290, page 1296. Citeseer.

Basaran, C., Yoon, H. J., Ra, H. K., Son, S. H., Park, T., and Ko, J.

(2014). Classifying children with 3d depth cameras for enabling children's safety applications. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 343–347.

Bobick, A. F. and Davis, J. W. (2001). The recognition of human movement using temporal templates. *IEEE Transactions on pattern analysis and machine intelligence*, 23(3):257–267.

Borges, P. V. K., Conci, N., and Cavallaro, A. (2013). Video-based human behavior understanding: A survey. *IEEE transactions on circuits and systems for video technology*, 23(11):1993–2008.

Boufama, B., Habashi, P., and Ahmad, I. S. (2017). Trajectory-based human activity recognition from videos. In *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pages 1–5. IEEE.

Boughorbel, S., Breebaart, J., Bruekers, F., Flinsenberg, I., and Ten Kate, W. (2010). Child-activity recognition from multi-sensor data. In *Proceedings of the 7th International Conference on Methods and Techniques in Behavioral Research*, pages 1–3.

Caba Heilbron, F., Escorcia, V., Ghanem, B., and Carlos Niebles, J. (2015). Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970.

Carreira, J., Noland, E., Banki-Horvath, A., Hillier, C., and Zisserman, A. (2018). A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*.

Carreira, J. and Zisserman, A. (2017). Quo vadis, action recognition? a new

model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308.

Castellano, B. (2017). Pyscenedetect.

Chambers, C., Seethapathi, N., Saluja, R., Loeb, H., Pierce, S. R., Bogen, D. K., Prosser, L., Johnson, M. J., and Kording, K. P. (2020). Computer vision to automatically assess infant neuromotor risk. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 28(11):2431–2442.

Chapel, M.-N. and Bouwmans, T. (2020). Moving objects detection with a moving camera: A comprehensive review. *Computer Science Review*, 38:100310.

Chen, B.-C., Chen, C.-S., and Hsu, W. H. (2014). Cross-age reference coding for age-invariant face recognition and retrieval. In *European conference on computer vision*, pages 768–783. Springer.

Chen, H.-S., Chen, H.-T., Chen, Y.-W., and Lee, S.-Y. (2006). Human action recognition using star skeleton. In *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pages 171–178.

Covre, N., Nunnari, F., Fornaser, A., and De Cecco, M. (2019). Generation of action recognition training data through rotoscoping and augmentation of synthetic animations. In *International Conference on Augmented Reality, Virtual Reality and Computer Graphics*, pages 23–42. Springer.

Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague.

Dalal, N., Triggs, B., and Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. In *European conference on computer vision*, pages 428–441. Springer.

Davis, J. and Domingos, P. (2009). Deep transfer via second-order markov logic. In *Proceedings of the 26th annual international conference on machine learning*, pages 217–224.

Diba, A., Sharma, V., and Van Gool, L. (2017). Deep temporal linear encoding networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2329–2338.

Dollár, P., Rabaud, V., Cottrell, G., and Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. (2015). Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634.

Efros, A. A., Berg, A. C., Mori, G., and Malik, J. (2003). Recognizing action at a distance. In *null*, page 726. IEEE.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Evgeniou, T. and Pontil, M. (2004). Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117.

Feichtenhofer, C., Fan, H., Malik, J., and He, K. (2019). Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211.

Feichtenhofer, C., Pinz, A., and Zisserman, A. (2016). Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941.

Fu, Y., Hospedales, T. M., Xiang, T., Gong, S., and Yao, Y. (2014). Interestingness prediction by robust learning to rank. In *European conference on computer vision*, pages 488–503. Springer.

Gao, J., Fan, W., Jiang, J., and Han, J. (2008). Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 283–291.

Ghadiyaram, D., Tran, D., and Mahajan, D. (2019). Large-scale weakly-supervised pre-training for video action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12046–12055.

Guha, T. and Ward, R. K. (2011). Learning sparse representations for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 34(8):1576–1588.

Guo, J., He, H., He, T., Lausen, L., Li, M., Lin, H., Shi, X., Wang, C., Xie, J., Zha, S., et al. (2020). Gluoncv and gluonnlp: Deep learning in computer vision and natural language processing. *Journal of Machine Learning Research*, 21(23):1–7.

Gupta, S. and Mooney, R. J. (2009). Using closed captions to train activity recognizers that improve video retrieval. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 30–37. IEEE.

Harris, C. G., Stephens, M., et al. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer.

Herath, S., Harandi, M., and Porikli, F. (2017). Going deeper into action recognition: A survey. *Image and vision computing*, 60:4–21.

Hesse, N., Bodensteiner, C., Arens, M., Hofmann, U. G., Weinberger, R., and Sebastian Schroeder, A. (2018). Computer vision for medical infant motion analysis: State of the art and rgb-d data set. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0.

Hesse, N., Schroeder, A. S., Müller-Felber, W., Bodensteiner, C., Arens, M., and Hofmann, U. G. (2017). Markerless motion analysis for early detection of infantile movement disorders. In *EMBEC & NBC 2017*, pages 197–200. Springer.

Hongeng, S. and Nevatia, R. (2003). Large-scale event detection using semi-hidden markov models. In *Computer Vision, IEEE International Conference on*, volume 3, pages 1455–1455. IEEE Computer Society.

Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics.

Hu, J.-F., Zheng, W.-S., Ma, L., Wang, G., Lai, J., and Zhang, J. (2018). Early action prediction by soft regression. *IEEE transactions on pattern analysis and machine intelligence*, 41(11):2568–2583.

Ince, O. F., Ince, I. F., Park, J.-S., Song, J.-K., and Yoon, B.-W. (2017). Child and adult classification using biometric features based on video analytics. In *ICIC International*, pages 819–825.

Ince, O. F., Park, J., Song, J., and Yoon, B. (2014). Child and adult classification using ratio of head and body heights in images. *International Journal of Computer and Communication Engineering*, 3(2):120.

Jaakkola, T. S., Haussler, D., et al. (1999). Exploiting generative models in discriminative classifiers. *Advances in neural information processing systems*, pages 487–493.

Jebara, T. (2004). Multi-task feature and kernel selection for svms. In *Proceedings of the twenty-first international conference on Machine learning*, page 55.

Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE.

Ji, S., Xu, W., Yang, M., and Yu, K. (2012). 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231.

Jiang, J. and Zhai, C. (2007). Instance weighting for domain adaptation in nlp. ACL.

Jiang, Y.-G., Dai, Q., Xue, X., Liu, W., and Ngo, C.-W. (2012). Trajectory-based modeling of human actions with motion reference points. In *European Conference on Computer Vision*, pages 425–438. Springer.

Kar, A., Rai, N., Sikka, K., and Sharma, G. (2017). Adascan: Adaptive scan pooling in deep convolutional neural networks for human action

recognition in videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3376–3385.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732.

Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al. (2017). The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*.

Kellokumpu, V., Zhao, G., and Pietikäinen, M. (2008). Human activity recognition using a dynamic texture based method. In *BMVC*, volume 1, page 2.

Khan, A., Sohail, A., Zahoora, U., and Qureshi, A. S. (2020). A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516.

Kim, E., Helal, S., and Cook, D. (2009). Human activity recognition and pattern discovery. *IEEE pervasive computing*, 9(1):48–53.

Klaser, A., Marszałek, M., and Schmid, C. (2008). A spatio-temporal descriptor based on 3d-gradients. In *BMVC 2008-19th British Machine Vision Conference*, pages 275–1. British Machine Vision Association.

Kong, Y. and Fu, Y. (2017). Max-margin heterogeneous information machine for rgb-d action recognition. *International Journal of Computer Vision*, 123(3):350–371.

Kong, Y. and Fu, Y. (2018). Human action recognition and prediction: A survey. *arXiv preprint arXiv:1806.11230*.

Kovashka, A. and Grauman, K. (2010). Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2046–2053. IEEE.

Laptev, I. (2005). On space-time interest points. *International journal of computer vision*, 64(2-3):107–123.

Laptev, I., Marszalek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Lawrence, N. D. and Platt, J. C. (2004). Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning*, page 65.

Lee, S.-I., Chatalbashev, V., Vickrey, D., and Koller, D. (2007). Learning a meta-level prior for feature relevance from multiple related tasks. In *Proceedings of the 24th international conference on Machine learning*, pages 489–496.

Lei, Q., Du, J.-X., Zhang, H.-B., Ye, S., and Chen, D.-S. (2019). A survey of vision-based human action evaluation methods. *Sensors*, 19(19):4129.

Li, J., Lin, D., Wang, Y., Xu, G., Zhang, Y., Ding, C., and Zhou, Y. (2020). Deep discriminative representation learning with attention map for scene classification. *Remote Sensing*, 12(9):1366.

Li, Q., Qiu, Z., Yao, T., Mei, T., Rui, Y., and Luo, J. (2017). Learning hierarchical video representation for action recognition. *International Journal of Multimedia Information Retrieval*, 6(1):85–98.

Liao, X., Xue, Y., and Carin, L. (2005). Logistic regression with an auxil-

iary data source. In *Proceedings of the 22nd international conference on Machine learning*, pages 505–512.

Liu, J., Kuipers, B., and Savarese, S. (2011). Recognizing human actions by attributes. In *CVPR 2011*, pages 3337–3344. IEEE.

Liu, J., Shahroudy, A., Xu, D., and Wang, G. (2016). Spatio-temporal lstm with trust gates for 3d human action recognition. In *European conference on computer vision*, pages 816–833. Springer.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., and Zhang, G. (2015). Transfer learning using computational intelligence: A survey. *Knowledge-Based Systems*, 80:14–23.

Matikainen, P., Hebert, M., and Sukthankar, R. (2009). Trajectons: Action recognition through the motion analysis of tracked features. In *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*, pages 514–521. IEEE.

Messing, R., Pal, C., and Kautz, H. (2009). Activity recognition using the velocity histories of tracked keypoints. In *2009 IEEE 12th international conference on computer vision*, pages 104–111. IEEE.

Mihalkova, L., Huynh, T., and Mooney, R. J. (2007). Mapping and revising markov logic networks for transfer learning. In *Aaai*, volume 7, pages 608–614.

Mihalkova, L. and Mooney, R. J. (2008). Transfer learning by mapping with minimal target data. In *Proceedings of the AAAI-08 workshop on transfer learning for complex tasks*.

Montoliu, R., Martín-Félez, R., Torres-Sospedra, J., and Martínez-Usó, A. (2015). Team activity recognition in association football using a bag-of-words-based method. *Human movement science*, 41:165–178.

Morel, M., Kulpa, R., Sorel, A., Achard, C., and Dubuisson, S. (2016). Automatic and generic evaluation of spatial and temporal errors in sport motions.

Murray, N., Marchesotti, L., and Perronnin, F. (2012). Ava: A large-scale database for aesthetic visual analysis. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2408–2415. IEEE.

Nam, Y. and Park, J. W. (2013). Child activity recognition based on cooperative fusion model of a triaxial accelerometer and a barometric pressure sensor. *IEEE journal of biomedical and health informatics*, 17(2):420–426.

Niu, Z., Zhou, M., Wang, L., Gao, X., and Hua, G. (2016). Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4920–4928.

Norvig, P. R. and Intelligence, S. A. (2002). *A modern approach*. Prentice Hall Upper Saddle River, NJ, USA:.

Ojala, T., Pietikainen, M., and Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987.

Pan, S. J. and Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Patron-Perez, A., Reid, I., Patron, A., and Reid, I. (2007). A probabilistic framework for recognizing similar actions using spatio-temporal features. In *Bmvc*, pages 1–10. Citeseer.

Perronnin, F. and Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE.

Perronnin, F. and Larlus, D. (2015). Fisher vectors meet neural networks: A hybrid classification architecture. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3743–3752.

Pirsiavash, H., Vondrick, C., and Torralba, A. (2014). Assessing the quality of actions. In *European Conference on Computer Vision*, pages 556–571. Springer.

Poppe, R. (2010). A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Rajagopalan, S., Dhall, A., and Goecke, R. (2013). Self-stimulatory behaviours in the wild for autism diagnosis. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 755–761.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Ricanek, K. and Tesafaye, T. (2006). Morph: A longitudinal image database of normal adult age-progression. In *7th International Con-*

ference on Automatic Face and Gesture Recognition (FGR06), pages 341–345. IEEE.

Sadanand, S. and Corso, J. J. (2012). Action bank: A high-level representation of activity in video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1234–1241. IEEE.

Sarabu, A. and Santra, A. K. (2020). Distinct two-stream convolutional networks for human action recognition in videos using segment-based temporal modeling. *Data*, 5(4):104.

Schroff, F., Criminisi, A., and Zisserman, A. (2010). Harvesting image databases from the web. *IEEE transactions on pattern analysis and machine intelligence*, 33(4):754–766.

Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360.

Sigurdsson, G. A., Gupta, A., Schmid, C., Farhadi, A., and Alahari, K. (2018). Charades-ego: A large-scale dataset of paired third and first person videos. *arXiv preprint arXiv:1804.09626*.

Sigurdsson, G. A., Varol, G., Wang, X., Farhadi, A., Laptev, I., and Gupta, A. (2016). Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer.

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. *arXiv preprint arXiv:1406.2199*.

Smaira, L., Carreira, J., Noland, E., Clancy, E., Wu, A., and Zisserman, A. (2020). A short note on the kinetics-700-2020 human action dataset. *arXiv preprint arXiv:2010.10864*.

Soomro, K., Zamir, A. R., and Shah, M. (2012). Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

Sun, D., Roth, S., and Black, M. J. (2010). Secrets of optical flow estimation and their principles. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2432–2439. IEEE.

Sun, J., Wu, X., Yan, S., Cheong, L.-F., Chua, T.-S., and Li, J. (2009). Hierarchical spatio-temporal context modeling for action recognition. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2004–2011. IEEE.

Suzuki, S., Mitsukura, Y., Igarashi, H., Kobayashi, H., and Harashima, F. (2012). Activity recognition for children using self-organizing map. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pages 653–658. IEEE.

Tang, K., Fei-Fei, L., and Koller, D. (2012). Learning latent temporal structure for complex event detection. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1250–1257. IEEE.

Tian, Y., Cao, L., Liu, Z., and Zhang, Z. (2011). Hierarchical filtered motion for action recognition in crowded videos. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(3):313–323.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M.

(2018). A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459.

Tu, Z., Xie, W., Qin, Q., Poppe, R., Veltkamp, R. C., Li, B., and Yuan, J. (2018). Multi-stream cnn: Learning representations based on human-related regions for action recognition. *Pattern Recognition*, 79:32–43.

Vrigkas, M., Nikou, C., and Kakadiaris, I. A. (2015). A review of human activity recognition methods. *Frontiers in Robotics and AI*, 2:28.

Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2013). Dense trajectories and motion boundary descriptors for action recognition. *International journal of computer vision*, 103(1):60–79.

Wang, H., Oneata, D., Verbeek, J., and Schmid, C. (2016a). A robust and efficient video representation for action recognition. *International journal of computer vision*, 119(3):219–238.

Wang, H. and Schmid, C. (2013). Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558.

Wang, H., Ullah, M. M., Klaser, A., Laptev, I., and Schmid, C. (2009). Evaluation of local spatio-temporal features for action recognition. In *Bmvc 2009-british machine vision conference*, pages 124–1. BMVA Press.

Wang, K., Wang, X., Lin, L., Wang, M., and Zuo, W. (2014). 3d human activity recognition with reconfigurable convolutional neural networks. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 97–106.

Wang, L., Qiao, Y., and Tang, X. (2015). Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4305–4314.

Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., and Van Gool, L. (2016b). Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer.

Wang, Y., Huang, K., and Tan, T. (2007). Human activity recognition based on r transform. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

Wang, Y. and Mori, G. (2010). Hidden part models for human action recognition: Probabilistic versus max margin. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1310–1323.

Weinland, D., Ronfard, R., and Boyer, E. (2006). Free viewpoint action recognition using motion history volumes. *Computer vision and image understanding*, 104(2-3):249–257.

Willems, G., Tuytelaars, T., and Van Gool, L. (2008). An efficient dense and scale-invariant spatio-temporal interest point detector. In *European conference on computer vision*, pages 650–663. Springer.

Wu, Z., Wang, X., Jiang, Y.-G., Ye, H., and Xue, X. (2015). Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 461–470.

Yeffet, L. and Wolf, L. (2009). Local trinary patterns for human action

recognition. In *2009 IEEE 12th international conference on computer vision*, pages 492–497. IEEE.

Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., and Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702.

Zhang, B., Wang, L., Wang, Z., Qiao, Y., and Wang, H. (2018). Real-time action recognition with deeply transferred motion vector cnns. *IEEE Transactions on Image Processing*, 27(5):2326–2339.

Zhang, H.-B., Zhang, Y.-X., Zhong, B., Lei, Q., Yang, L., Du, J.-X., and Chen, D.-S. (2019). A comprehensive survey of vision-based human action recognition methods. *Sensors*, 19(5):1005.

Zhang, Z., Song, Y., and Qi, H. (2017). Age progression/regression by conditional adversarial autoencoder. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5810–5818.

Zhao, G. and Pietikainen, M. (2007). Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):915–928.

Zhao, H., Torralba, A., Torresani, L., and Yan, Z. (2019). Hacs: Human action clips and segments dataset for recognition and temporal localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8668–8678.

Ziaeefard, M. and Bergevin, R. (2015). Semantic human activity recognition: A literature review. *Pattern Recognition*, 48(8):2329–2345.

Appendices

Appendix A

Sport Labels

1. Archery
2. Bouncing on trampoline
3. Bowling
4. Cartwheeling
5. Catching or throwing frisbee
6. Catching or throwing baseball
7. catching or throwing softball
8. Dribbling basketball
9. Dunking basketball
10. Hitting baseball
11. Juggling soccer ball
12. Kicking soccer ball
13. Parkour

14. Playing badminton
15. Playing basketball
16. Playing cricket
17. Playing tennis
18. Playing volleyball
19. Shooting basketball
20. Somersaulting
21. Water skiing

Appendix B

Experimental Results

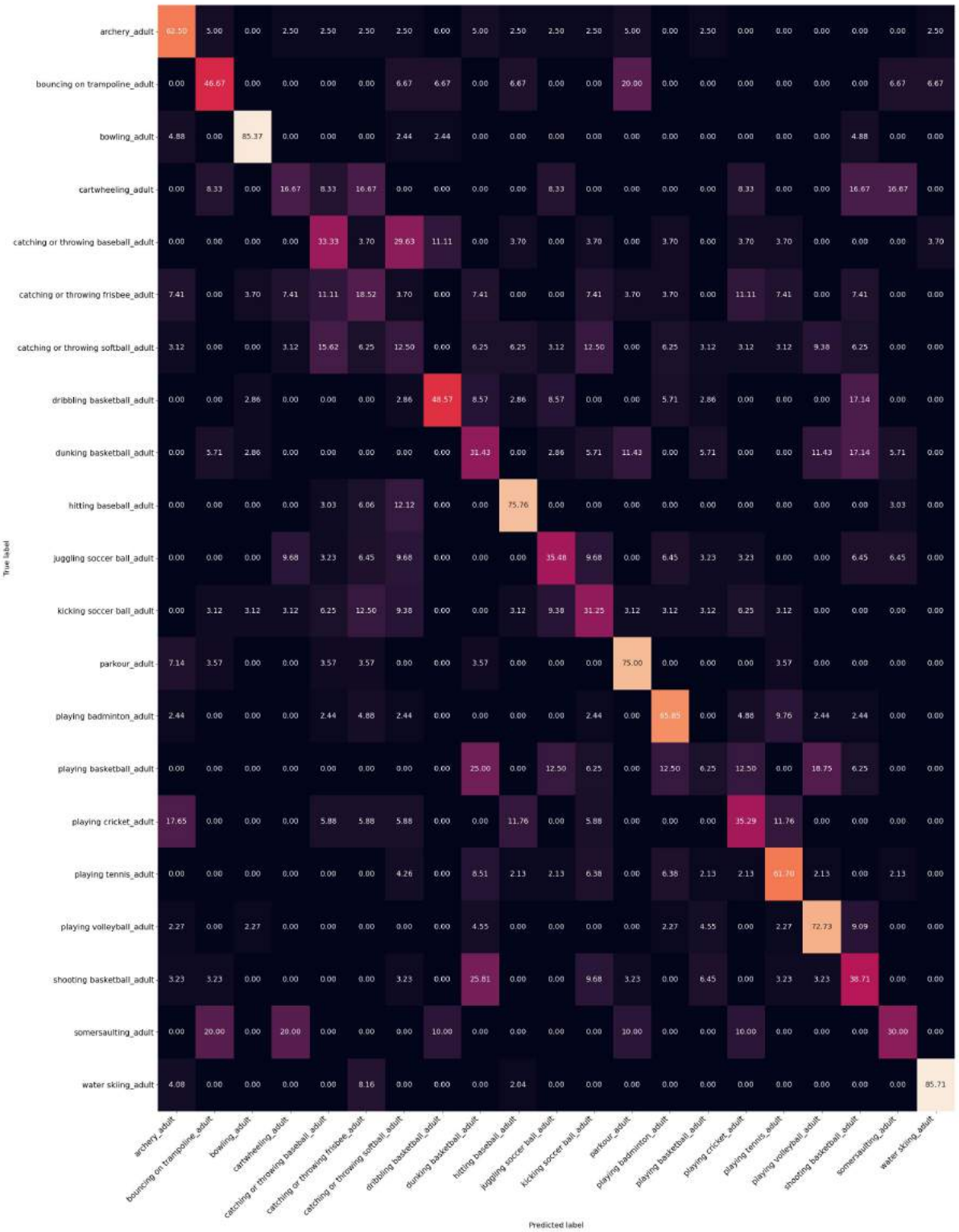

Figure B.1: Confusion matrix of Adult-specific model evaluated on adult-specific dataset.

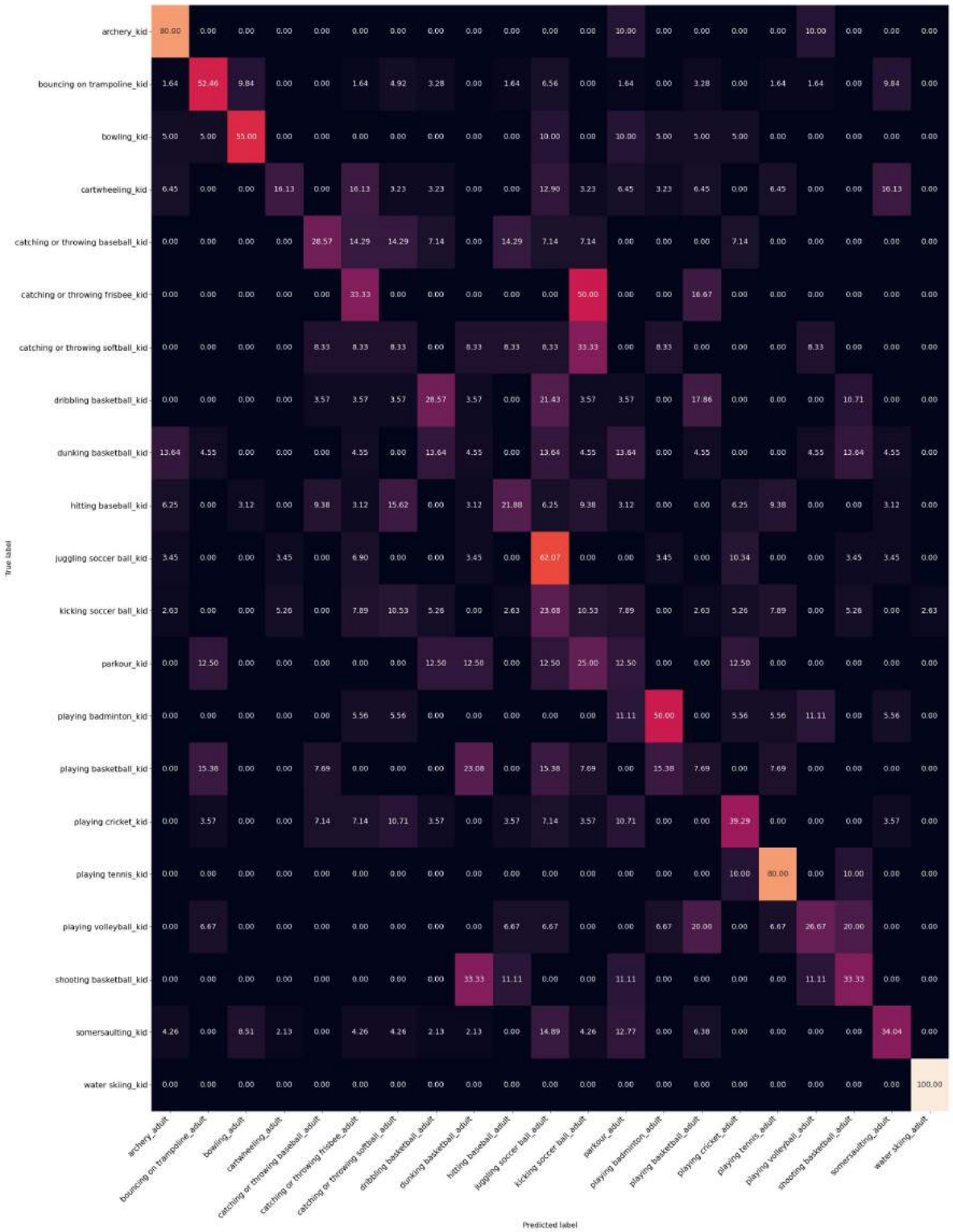

Figure B.2: Confusion matrix of Adult-specific model evaluated on kid-specific dataset.

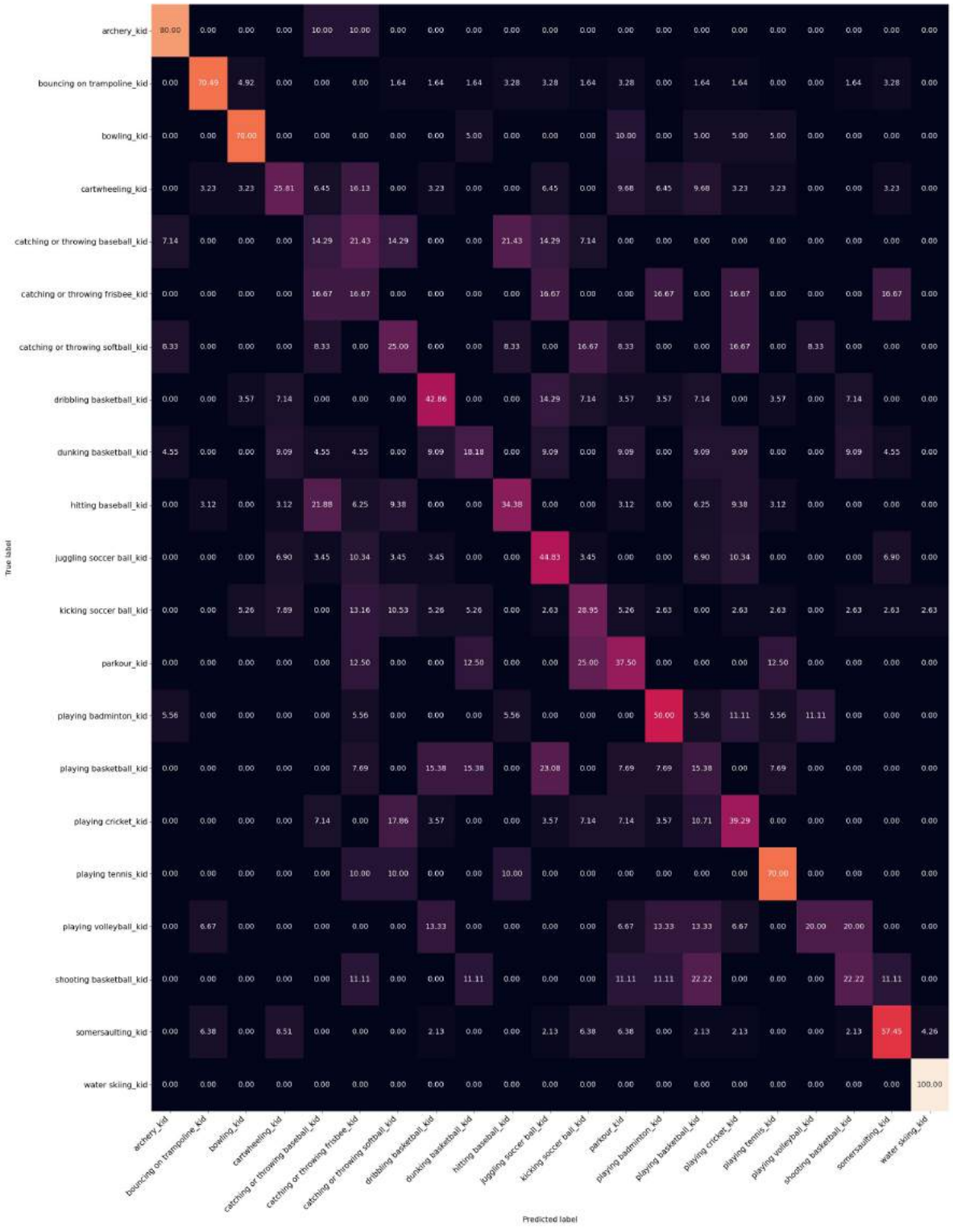

Figure B.3: Confusion matrix of Kid-specific model evaluated on kid-specific dataset.

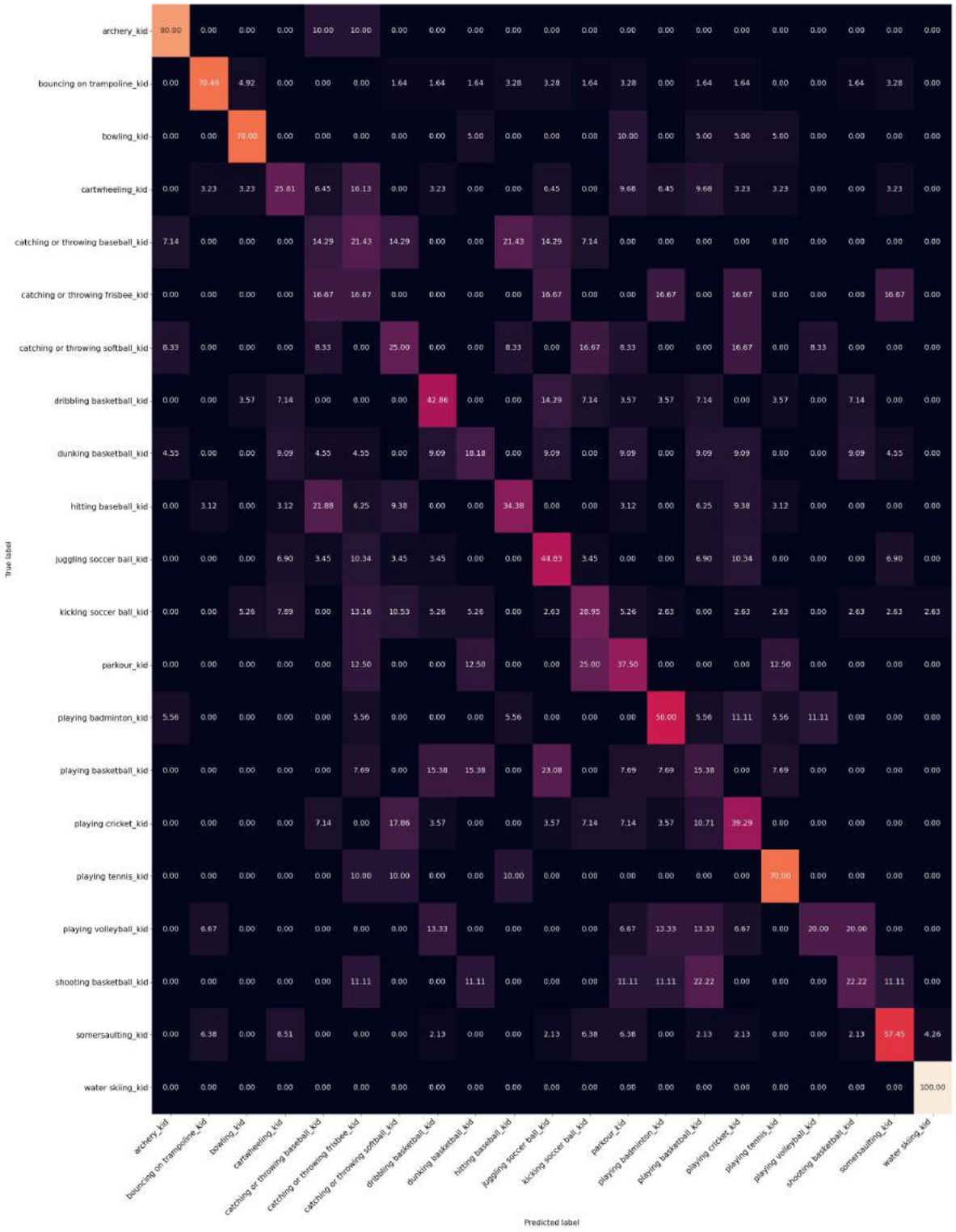

Figure B.4: Confusion matrix of Kid-specific model evaluated on adult-specific dataset.

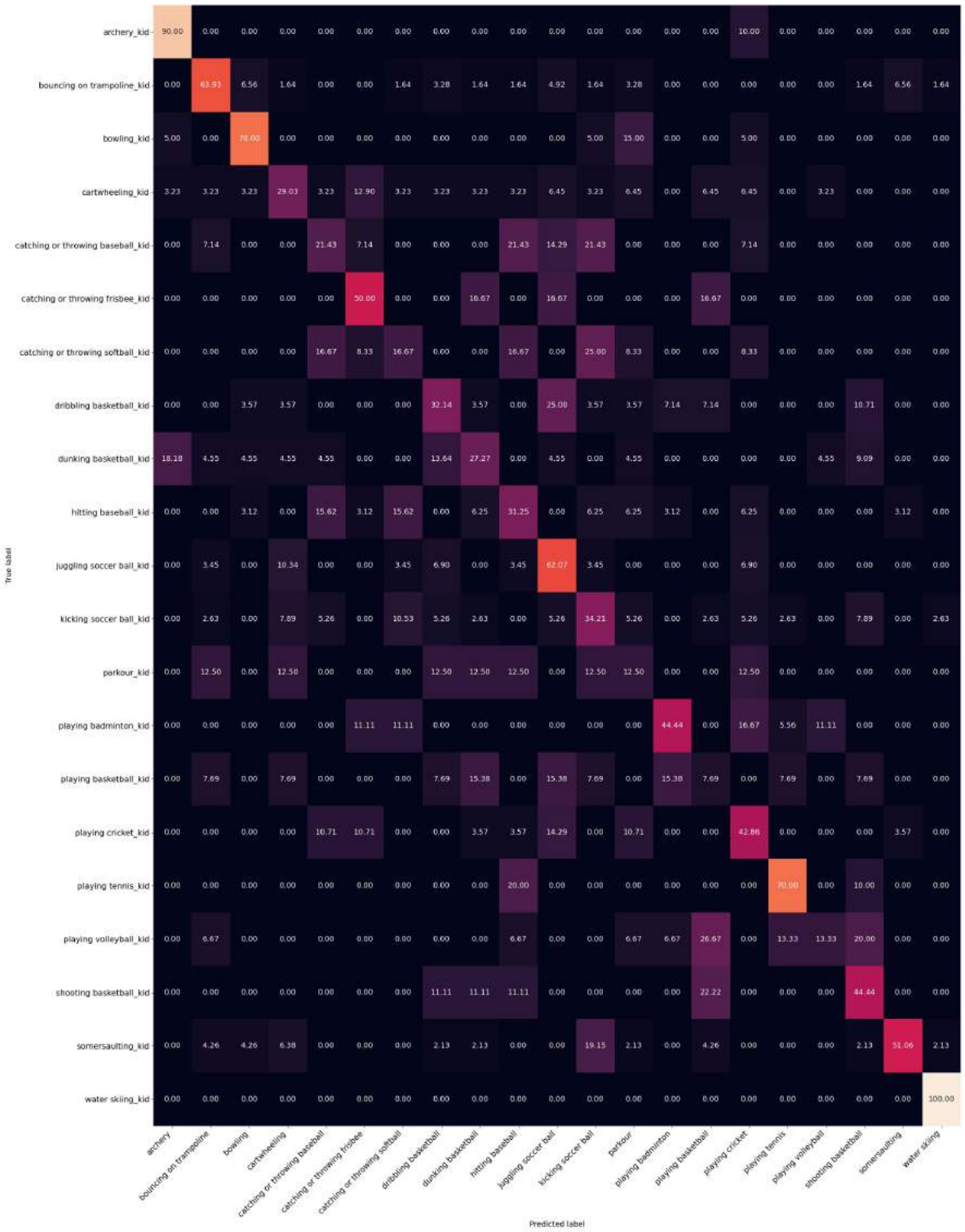

Figure B.5: Confusion matrix of Mixed model (half-split) model evaluated on kid-specific dataset.

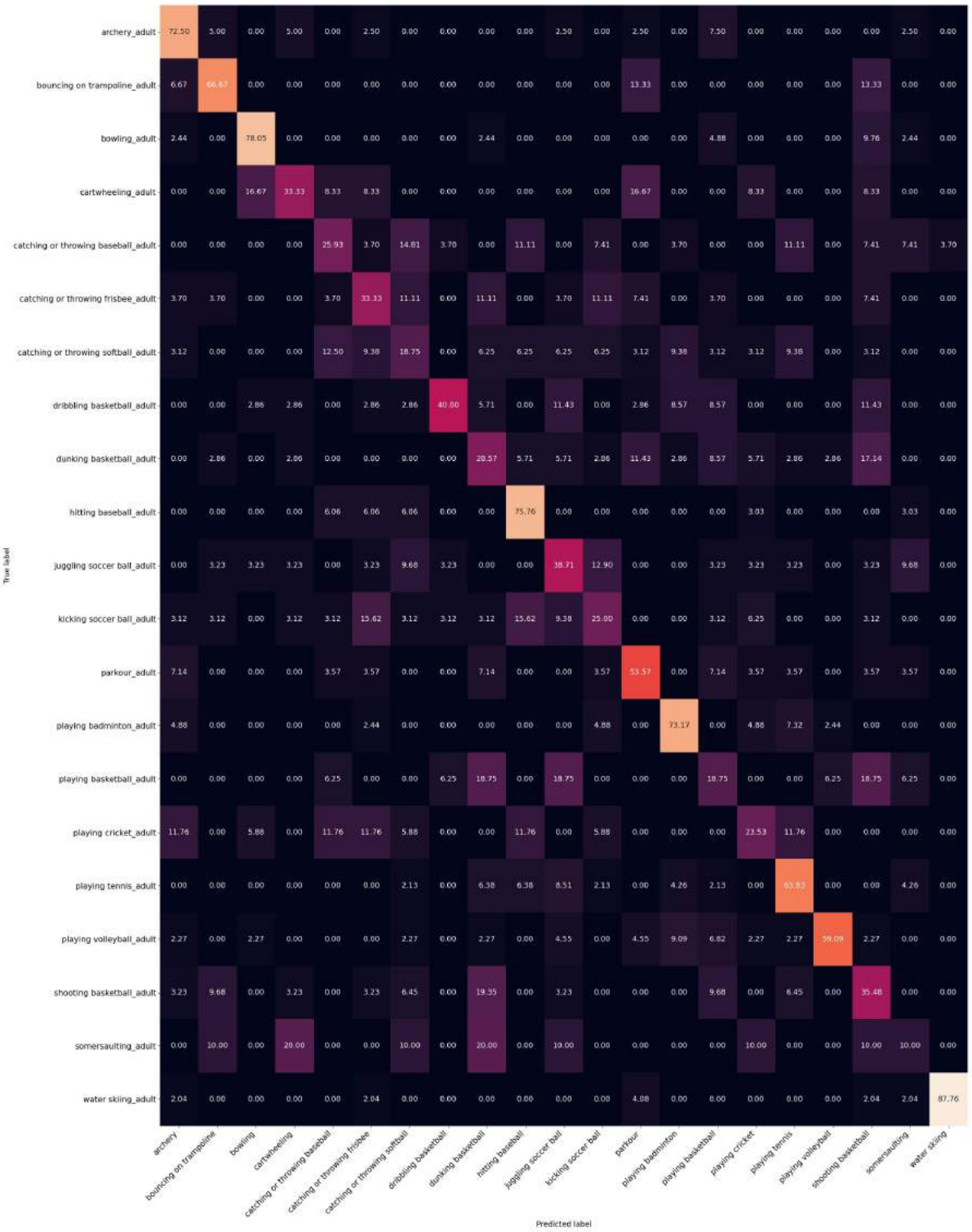

Figure B.6: Confusion matrix of Mixed model (half-split) model evaluated on adult-specific dataset.

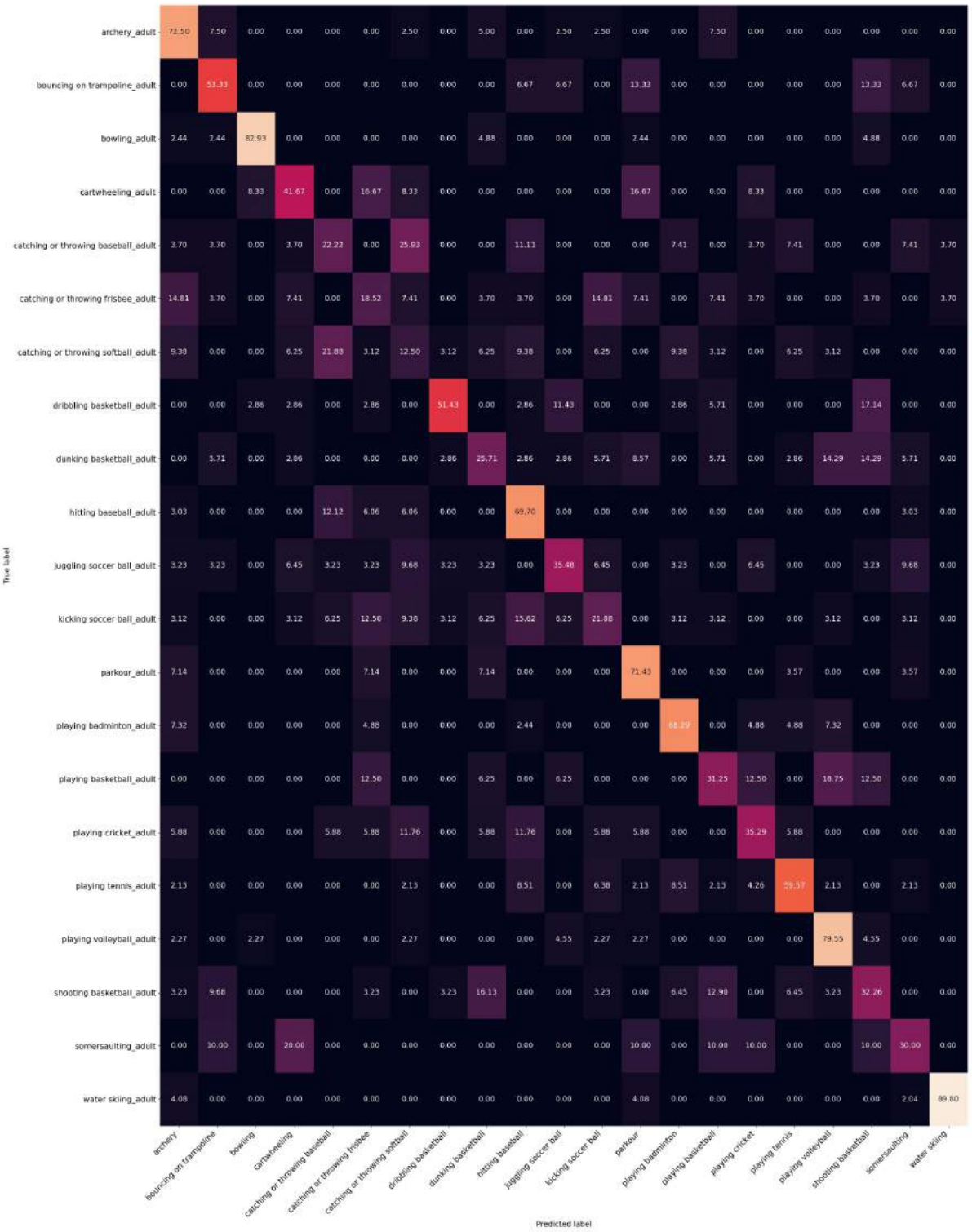

Figure B.7: Confusion matrix of Mixed model (full-split) model evaluated on adult-specific dataset.

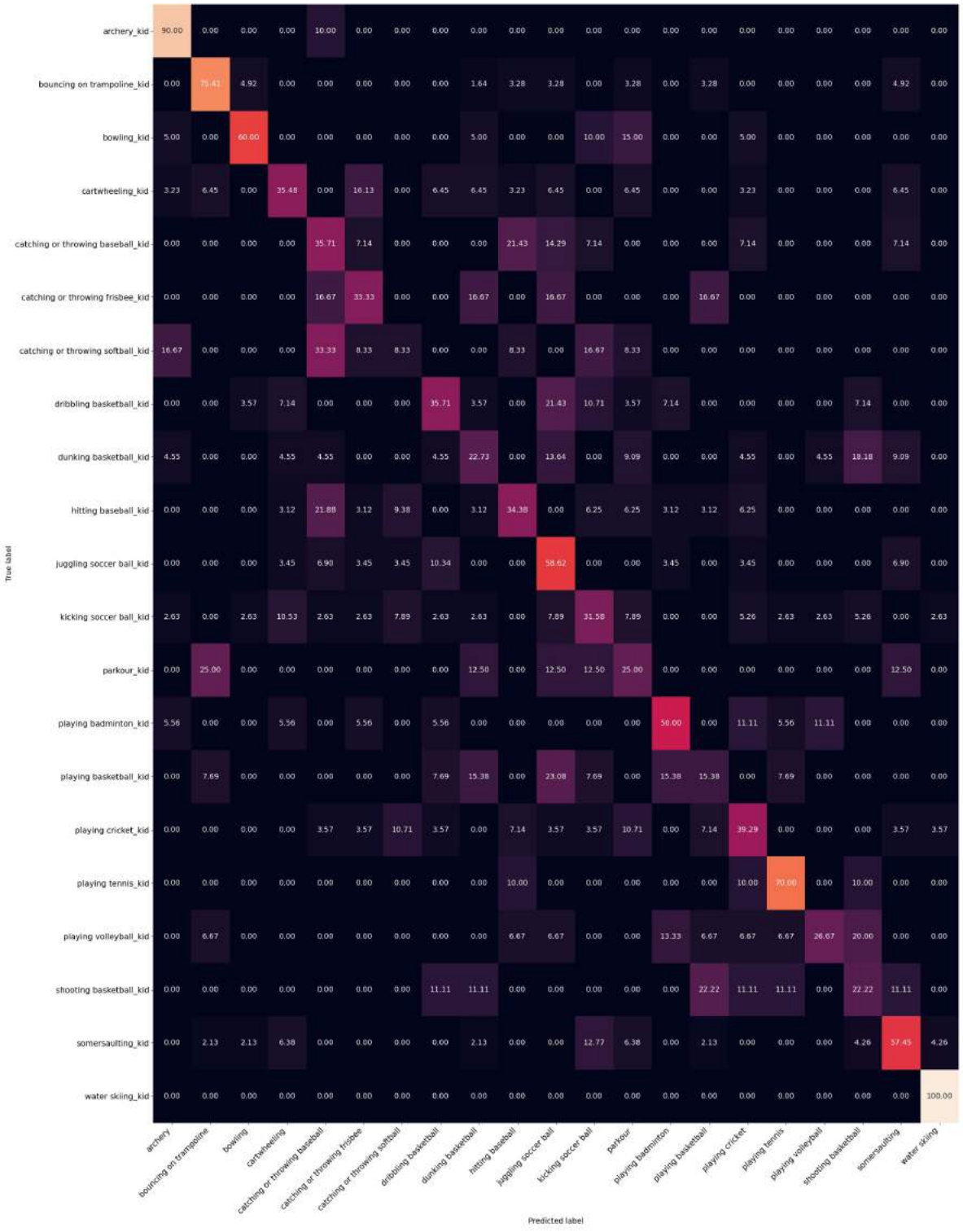

Figure B.8: Confusion matrix of Mixed model (full-split) model evaluated on kid-specific dataset.