Hand Guided High Resolution Feature Enhancement for Fine-Grained Atomic Action Segmentation within Complex Human Assemblies

Matthew Kent Myers
School of Engineering
Newcastle University
M.J.Kent-Myers2@ncl.ac.uk

Nick Wright
School of Engineering
Newcastle University

A. Stephen McGough
School of Computing
Newcastle University

Nicholas Martin
Tharsus Ltd.

Abstract

Due to the rapid temporal and fine-grained nature of complex human assembly atomic actions, traditional action segmentation approaches requiring the spatial (and often temporal) down sampling of video frames often loose vital fine-grained spatial and temporal information required for accurate classification within the manufacturing domain. In order to fully utilise higher resolution video data (often collected within the manufacturing domain) and facilitate real time accurate action segmentation – required for human robot collaboration – we present a novel hand location guided high resolution feature enhanced model. We also propose a simple yet effective method of deploying offline trained action recognition models for real time action segmentation on temporally short fine-grained actions, through the use of surround sampling while training and temporally aware label cleaning at inference. We evaluate our model on a novel action segmentation dataset containing 24 (+background) atomic actions from video data of a real world robotics assembly production line. Showing both high resolution hand features as well as traditional frame wide features improve fine-grained atomic action classification, and that though temporally aware label clearing our model is capable of surpassing similar encoder/decoder methods, while allowing for real time classification.
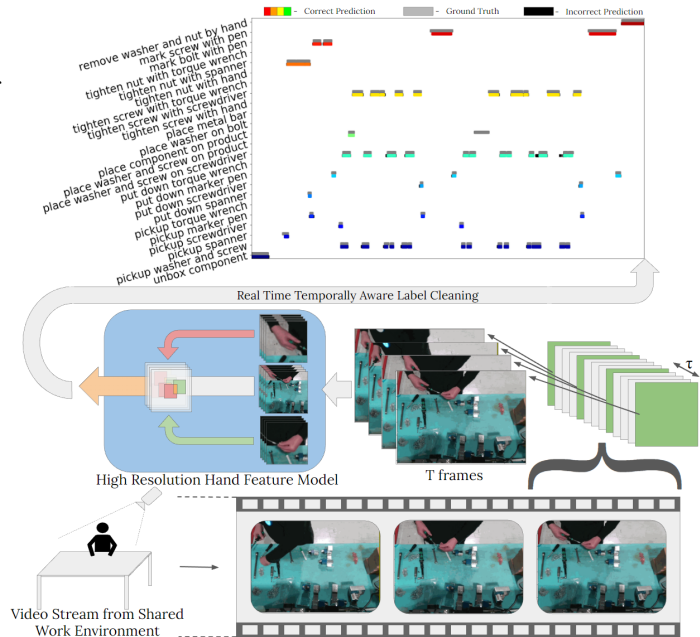
Figure 1: Overview of our action segmentation approach to classifying real world assembly data. We operate temporally aware label cleaning in an overlapping sliding window fashion, utilising a temporally aware high resolution hand feature enhanced frame wide backbone model.

1. Introduction

While advances in robotics have made significant progress in automating many simple and repetitive labour heavy tasks within manufacturing, many complex manufacturing assemblies requiring assembly adaptability or robustness still require human operators. Human-Robot Collaboration (HRC) has been identified as a key component in moving towards a smart assembly procedure in the next few years [41]. Where humans and robots can work in tandem on assemblies, leading to increased productivity, reliability, more attractive working conditions [38], as well create a plethora of quality assurance and guidance tools

(*e.g.* [10, 11]) reducing human error and speeding up assembly. One of the core challenges of implementing successful HRC systems is the ability for machines to understand the environment around them in a quick and robust manner; with a successful HRC system requiring the ability to understand an assembler's actions and intentions in real time in order to intelligently assist them and provide an efficient hybrid work system [26].

Motivated by recent atomic-action datasets (*e.g.* [3, 4, 20, 29]) and the significant advances in fine-grained action recognition (AR) the utilisation of video data captured within the shared work environment is now an attractive, unintrusive, and information rich approach to classifying

fundamental atomic actions between a worker, product, part or tool. With the accurate classification of these actions allowing assemblies to be understood as a nested set of fundamental atomic actions. Whilst many works have focused on classifying actions within manufacturing environments, most of these works have been limited in their approach; often operating within a controlled environment, classifying coarse actions with single frames, or performing action segmentation in an offline manner. In contrast, many HRC systems require real time classification of spatially and temporally similar fine-grained actions, which is often very challenging due to the spatial limitations of AR inputs (224x224) where often significant downsampling is used – removing vital information collected by the camera.

With this motivation in mind we develop a novel AR model facilitating the use of enhanced spatial features via hand guided high resolution feature enhancement, capable of operating in real time in a sliding window fashion with temporally aware label cleaning for action segmentation, as shown in Fig. 1. Our novel model architecture is evaluated on a domain specific novel assembly dataset collected from a real world robotics sub assembly manufacturing production line. We perform extensive experimentation on extended sequences of unseen video data and provide a class specific performance analysis, revealing the importance of enhanced hand features for improving fine-grained atomic action classification. Our main contributions are as follows:

- We Introduce a novel AR model utilising hand location guided high resolution feature extraction to allow fine grained details to aid a backbone AR model and show both high resolution and frame wide features are key to accurate classification.
- We show that running a temporally aware model trained on short video clips from our domain dataset can be successfully applied in a simple sliding window via the use of surround sampling during training and temporally aware label cleaning, surpassing the performance of similar encoder/decoder methods, whilst operating in real time.
- We show that the performance of encoder/decoder methods can be improved significantly, and training time reduced, via the use of in domain feature extraction via model training under our framework.

2. Related Work

2.1. Action Segmentation

Current action segmentation approaches can be broadly broken down into two stages: An initial feature extraction process (often on a frame or short segment level) followed by a frame-wise sequence prediction stage, often facilitating the ability to learn longer term temporal dependencies to refine action boundaries and reduce over segmentation.

2.1.1 Feature Extraction

Feature extraction generally falls into two main categories: 2D or 3D Convolutional Neural Network (CNN) based methods. 2D CNN methods utilise a 2D CNN [13, 19, 34] backbone to extract frame level predictions with a temporal aggregation step, such as a simple averaging [39] or a more complex operation such as an LSTM block [5] to produce features for a set of input frames. In contrast, 3D CNN based methods (*e.g.* [12, 25, 33, 36, 37]) expand the 2D convolutional operation into the temporal dimension, performing 3D convolutions across space and time, extracting joint spatial and temporal semantics from voxels of raw video data. Other works utilise the inherent computational efficiency of 2D CNNs via the incorporation of temporally aware modules inserted within 2D CNN architectures (*e.g.* [22, 32]) facilitating the mixing of features across frames, allowing for a computationally efficient and parameter light methods of facilitating temporal learning.

Our work is similar in motivation to many works focusing on improving feature extraction via lateral connections between different layers of a model [23, 43], different models operating on different frame rates [8] or modalities [9]. In contrast to these approaches which all utilise a single fixed input resolution, we utilise a secondary stream with access to higher resolution image information which is used to enhance the backbone frame wide features.

2.1.2 Sequence Predictions

In order to produce frame level predictions across a long untrimmed video many temporal modelling methods have been used. Most simply, a sliding window [27], often operating with a recurrent network [30] or a feature buffer [18, 22] to allow for arbitrary length temporal modelling in an online manner, can be utilised to make sequence predictions. Many approaches replace the relatively short term temporal modelling capabilities of recurrent models with temporal convolutions [2], improving action boundaries and minimising over segmentation errors via encoder/decoder methods [6, 21] typically operating on precomputed features extracted by a pretrained 3D CNN model. In contrast, we show show through the use of surround sampling while training on in domain data, a simple temporally aware label cleaning method at inference can outperform more computationally expensive encoder/decoder methods operating on kinetics [16] pretrained features, while performing real time action segmentation.

2.2. Assembly Understanding

Since the breakthrough of the CNN during the 2010s, there has been fairly limited work applying AR models to the manufacturing domain in order to understand and facilitate HRC. Wang *et al.* [40] fine-tune two AlexNet[19] models to classify three parts being used as well as three current "motions" (grasping, holding and assembling) during an engine block assembly within a controlled environ-
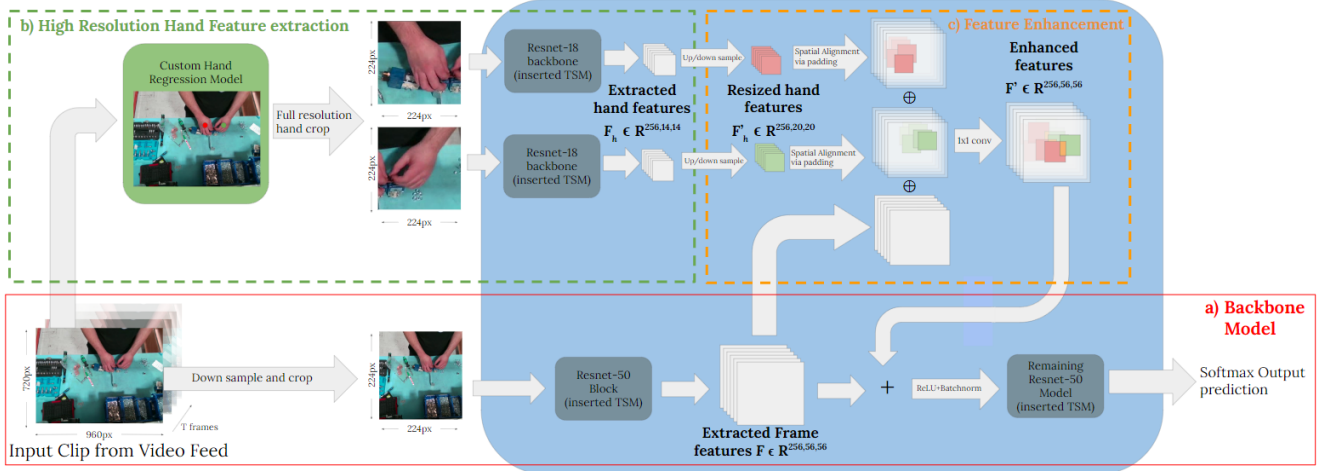
Figure 2: An overview of our proposed method of utilising high resolution hand features to improve frame level features. Our method can be broken into three main sections: a) A backbone model extracting frame wide features from a set of spatially downsampled video frames. b) Feature extraction from higher resolution images focused on locations extracted via a hand regression model. c) Backbone model feature enhancement via the spatially aware combination of extracted features.

ment. Al-Amin *et al.* [1] utilise a multi-modal sensor fusion, including image data, to classify seven actions within a controlled assembly process. Zhang *et al.* [44] utilise a bi-stream CNN to extract human and object information from video frames, with a variable-length Markov model making future action predictions within a controlled assembly scenario. Xiong *et al.* [42] incorporates motion information via the use of a modest two stream network classifying actions occurring within an engine block assembly video scraped from YouTube. Jones *et al.* [15] develop a framework for understanding fine-grained assembly actions as a structure. In contrast to these works, which are all limited in approach, and are deployed within controlled environments, our work utilises fully spatio-temporal feature extraction and is applied to our novel real world assembly dataset.

As human assemblies naturally utilise hands, there are several works with similar motivations to ours. Lui *et al.* [24] utilise a modest 3D CNN operating on fixed length segments composed of cropped hand images to classify real world assembly data into seven classes. Kobayashi *et al.* [17] implement a similar method however note that such inputs remove much of the contextual surrounding information which is often required for accurate classification. As such in the same work Kobayashi *et al.* introduce an attention based model where pose features from single video frames are used to draw attention to hands on real world assembly data. Images are classified into eleven actions (such as take product, put on component, grab driver) using an encoder/decoder for action segmentation. In contrast, we show that using higher resolution pixel data from around the hands to enhance full frame features provides more information than drawing attention to hands during assemblies, whilst allowing for full frame features to provide context when classifying, unlike hand cropping methods.

3. Proposed Method

The following section introduces our proposed method for action segmentation within the manufacturing domain. Our method aims to specifically tackle the challenge of distinguishing between spatially fine-grained actions, where two temporally similar actions *e.g. tighten screw with torque wrench vs. tighten nut with torque wrench* require fine-grained spatial detail to separate. As the vast majority of AR approaches are based on image classification CNNs requiring a fixed size input of 224x224 pixels, high resolution video frames (typically upwards of 720p) must be downsampled before being input into the model, removing vital high resolution information required to distinguish between small or similar looking components/parts. In order to address this issue our method outlined in Fig. 2 utilises high resolution data extracted around the assembler's hands (where most relevant fine grained information: tool, component, fixings *etc.* is typically concentrated) to enhance the features extracted by a backbone frame wide model. Our model can be separated into 3 main section: a) a backbone frame wide feature extractor, b) high resolution hand feature extraction and c) back bone feature enhancement via a spatially aware feature enhancement module, discussed in section 3.1, 3.2 and 3.3 respectively.

3.1. Backbone Model

As shown in section (a) (highlighted red) of Fig. 2 we build upon a generic backbone action recognition model (see section 3.4.1 for specific model selection) to extract features from the frame level across the input clip and produce a final action classification. This backbone operates on spatially downsized video voxels of size $\mathbb{R}^{T,3,224,224}$ where

T is the number of 224x224 RGB image frames, producing frame level feature maps, $\mathcal{F} \in \mathbb{R}^{T,C,H,W}$, which we seek to enhance via high resolution hand features.

3.2. High Resolution Hand Feature Extraction

Shown in section (b) (highlighted green) of Fig. 2 the second section of our model is designed to extract features from around the hands in order to aid the classification of actions with fine-grained spatial detail and is a two stage process.

First, to detect the presence and location of an assemblers hand within an image we train a custom mobilenetv2 [28] model to predict the probability and location of hands within an image. As our model relies on the accurate localisation of an assemblers hands within video frames we report the high performance of our hand model in section 1 of the supplementary material along with model architecture, implementation and training details.

Second, a crop of size (h_c, w_c) is taken centred on hand locations retaining the resolution of the full input image[1]. We truncate the hand models to extract features from an intermediate layer, producing two hand feature maps, $\mathcal{F}_h \in \mathbb{R}^{T,C',H',W'}$ in order to retain spatial features and limit parameters used within these network streams (see section 3.4 where we discuss our specific model and layer choices).

3.3. Feature Enhancement

At the core to our model is the feature enhancement stage whereby extracted hand features maps ($\mathcal{F}_h$) are used to enhance the backbone model features ($\mathcal{F}$). We introduce the feature enhancement strategy highlighted by (c) (highlighted yellow) in Fig. 2 providing the models and dimensions of feature maps we select in section 3.4. Our feature enhancement strategy allows for the spatial alignment and combination of features of any size from the hand stream to the features at any point in the backbone architecture.

3.3.1 Spatial Alignment

Due to the higher relative resolution of the hand input, compared to the frame wide input image, the hand input image and all hand feature maps ($\mathcal{F}_h$) extracted by the hand model correspond to a fixed region within the larger input frame and all subsequent feature maps $\mathcal{F}$ of the backbone model. In order to account for the large discrepancy in spatial input space between the hand and backbone stream, and to combine features in a spatially consistent manner we propose a spatial alignment process whereby features extracted from the high resolution hand data are mapped to their corresponding size and locations with respect to the input image via the use of up/down sampling zero padding. These padded feature maps represent the high resolution hand features with respect to the backbone architecture,

which can be used to enhance the spatial features of the backbone feature maps. In order to achieve this spatial alignment between hand cutouts of size (w_c, h_c) at location (x_{hand}, y_{hand}), and the backbone randomly cropped input of size C_s at an offset (x_c, y_c), we calculate the normalised size $(\overline{w}_{hand}, \overline{h}_{hand})$, and normalised offsets $(\overline{x}_{hand}, \overline{y}_{hand})$ with in equation 1 and 2 respectively. Where H is the height of full resolution frame and S is the height of the scaled input image before random cropping – a more detailed description and diagram are provided in section 2 of the supplementary material.

$$\overline{w}_{hand} = \left(\frac{w_c}{H}\right)\left(\frac{S}{C_s}\right), \quad \overline{h}_{hand} = \left(\frac{h_c}{H}\right)\left(\frac{S}{C_s}\right) \quad (1)$$

$$\overline{x}_{hand} = \frac{x_{hand} - x_c}{C_s}, \quad \overline{y}_{hand} = \frac{y_{hand} - y_c}{C_s} \quad (2)$$

Once the normalised offsets and size have been calculated, $\mathcal{F}_h$ is up/downsampled (using nearest neighbour when upsampling) to spatially match the size of the hand with respect to back bone feature map $\mathcal{F}$, to produce an intermediate feature map $\mathcal{F}'_h \in \mathbb{R}^{T,C,H*\overline{h},W*\overline{w}}$. We then zero pad around $\mathcal{F}'_h$ to match the size of $\mathcal{F}$ using the offsets $(\overline{x}_{hand}, \overline{y}_{hand})$ to place the features at the corresponding location within $\mathcal{F}$ producing a set of feature maps $\mathcal{F}' \in \mathbb{R}^{T,C,H,W}$ matching the backbone spatial dimensions whilst containing high resolution hand features spatially located at the position of hands within the backbone feature map $\mathcal{F}$.

3.3.2 Feature Combination

Once hand features have been spatially aligned using zero padding to match the spatial dimension of the full frame features, they are concatenated along the channel dimension, producing a set of separate but spatially matched backbone and high resolution hand features $\in \mathbb{R}^{T,3C,H,W}$.

To combine these separate features into one feature set of enhanced full frame feature maps a 1x1 convolution, of kernel size C is utilised to reduce the channel dimension from 3C back to that of the original extracted feature map $\mathcal{F}$. This convolution allows for spatially preserved channel mixing between the feature maps and produce a set of enhanced hand feature maps. The enhanced feature maps are then added to the original extracted frame features, $\mathcal{F}$, via a residual connection, followed by a linear activation function and batch normalisation.

3.4. Model & Feature Combination Stage Selection

Whilst our outlined method has so far has been invariant to model choice hand crop size and the intermediate feature maps selection, we now outline the choices made when implementing our model for the experiments described in section 4.

---

[1]If no hand is predicted by the model we input a central crop from the full image, which we found to perform better than a zeroed input.

3.4.1 Model Selection

Following convention we opt for a ResNet-50[13] backbone and ResNet-18 models for the two hand streams. However note other more efficient backbones *e.g.* [14] [35] could be utilised for real time operation on edge devices. We also select a hand crop size of $w_c = h_c = 224$ and the random spatial crop $C_s = 224$ as this is the standard input size for a ResNet model, and simplifies equation 1 to $\overline{w} = \overline{h} = \frac{S}{W}$.

We operate models across multiple frames utilising the Temporal Segment Network[39] approach of averaging frame levels predictions over an input clip. To facilitate the efficient learning of temporal features between frames, we utilise the Temporal Shift Module[22] which is inserted in every residual block for all models.

3.4.2 Combination Stage

When selecting the feature maps to utilise for feature combination, we note that minimal downsampling/upsampling is preferred when matching the two feature sets. As such we utilise the consecutive downsampling of the hand streams to the point where the downsampled hand features match the size of the hand within one of the early feature map within the backbone model (*i.e.* $H\overline{h} \approx H'$). We therefore opt to enhance the features from the first block of a ResNet-50 backbone ($\mathcal{F} \in \mathbb{R}^{T,256,56,56}$) with the features from a truncated hand models at the penultimate residual block ($\mathcal{F}_h \in \mathbb{R}^{T,256,14,14}$), as the backbone hands have a size of $H\frac{S}{H} = 56 * (\frac{256}{720}) \approx 20$ and $H' = 14$.

3.5. Real Time Action Segmentation

We extend our offline action recognition approach to action segmentation via a simple overlapping sliding window methodology. During inference we utilise a dense frame sampling strategy to artificially create input segments corresponding to the current video frame x_0. T frames are sampled at a temporal stride of τ going backwards from the most recently received video frame to create an input clip of fixed length $\{x_{-T\tau}, x_{-(T-1)\tau}, ..., x_{-\tau}, x_0\}$ which is used to create a classification for the middle frame of the clip, *i.e.* $x_{-T\tau/2}$.

3.5.1 Training Clip Creation - Surround Sampling

To match the dense sampling strategy utilised at inference, during training clips of length $T\tau$ must be sampled from examples within our dataset containing N frames. To ensure our model can accurately determine the start and end of actions in sequences within a video feed we implement a surround sampling procedure where the starting frame of the training sample (*i.e.* $x_{-T\tau}$) is chosen via equation 3, where N_s and N_e are the segment level labelled start and end frame of the segment within the dataset, T is the number of frames selected and τ is the temporal stride between frames.

$$x_{-T\tau} = U\{N_s - \frac{T\tau}{2}, N_e - \frac{T\tau}{2}\}, s.t. \ x_{-T\tau} \in \mathbb{N} \quad (3)$$

Importantly, this sampling procedure allows frames from outside the labelled segment to be seen by the model during training. As the model produces a prediction for the central frame of the segment this sampling strategy assists the model at accurately predicting the boundaries of actions much better than sampling strategy where training clips are created from data fixed within the bounds of an action.

During offline testing clips are sampled from the centre of the N frame examples, with the initial frame selected as $x_{-T\tau} = \frac{N_f - N_s}{2} - \frac{T\tau}{2}$.

3.5.2 Temporally Aware Label Cleaning

Under the outlined framework, our model is used to produce per frame classification when operating in an overlapping sliding window fashion on sequences of video data. We implement an additional temporally aware label cleaning stage to improve the model performance on sequences of data as a method of reducing over segmentation while retaining real time implementation rather than using an offline method such as encoder/decoder. While the model is running across a sequence of data any predictions which are statistically too short, that is have a length shorter than $C_{mean} - \kappa C_{std}$, where C_{mean} and C_{std} are the mean and standard deviation of the lengths of class C within the training data, and κ is a constant which we find by sweeping between $1 \rightarrow 2$ on the training data to determine the best cutoff for performance, are ignored by the model with the previous action prediction being taken as true.

4. Experimental Results

The following section presents the experimental results of our enhanced hand feature model on our newly collected atomic assembly dataset, comparing to baseline and similar works. We further discuss the deployment of our model (and others) for real time action segmentation through the use of temporally aware label cleaning, comparing the model to more computationally heavy encoder/decoder methods, highlighting the importance of short term temporal feature learning for fine-grained atomic action segmentation.

4.1. Atomic Assembly Dataset

In order to evaluate the effectiveness of our model for HRC tasks we create a novel manufacturing action segmentation dataset. A D415 Intel RealSense depth camera (15 fps @ 920x720) was used over 3 days to record 3 different workers complete a real world robotics assembly procedure. The camera was positioned above a workbench with the camera angled to capture the bench surface, workers hands and lower torso, avoiding discernible features. Assemblers

i.d	Interaction	Object	Secondary Object	N° of Examples	Mean Length (sec)	Total Length (min)
0	Unbox	Component	-	82	8.63	11.80
1		Washer and/or Screw	-	610	2.07	21.08
2		Spanner	-	50	1.44	1.20
3	Pick Up	Screwdriver	-	101	1.02	1.73
4		Marker Pen	-	149	1.12	2.78
5		Torque Wrench	-	85	1.14	1.62
6		Spanner	-	49	0.96	0.78
7	Put Down	Screwdriver	-	102	1.13	1.92
8		Marker Pen	-	140	1.27	2.96
9		Torque Wrench	-	81	1.07	1.49
10		Washer and Screw	on Screwdriver	428	1.86	13.29
11		Washer and Screw	on Product	159	1.78	4.72
12	Place	Component	on Product	123	3.43	7.03
13		Washer	on Bolt	74	2.76	3.40
14		Metal Bar	on Product	38	3.59	2.27
15		Screw	with Hand	163	3.80	10.32
16		Screw	with Screwdriver	359	5.32	31.85
17	Tighten	Screw	with Torque Wrench	168	3.39	9.49
18		Nut	with Hand	77	8.43	10.82
19		Nut	with Spanner	220	1.99	7.28
20		Nut	with Torque Wrench	95	7.74	12.26
21	Mark	Bolt	with Marker Pen	134	4.46	9.97
22		Screw	with Marker Pen	105	12.89	22.55
23	Remove	Washer and Nut	from Product	76	7.95	10.07
24	**No Action**	-	-	**1220**	**3.75**	**76.26**

Table 1: Summary of assembly dataset statistics.

Model	Segment Level F_1	Sequence Segmentation F_1@{0.5,0.25,0.1}		
Single Frame Baseline	68.6	30.7	44.9	51.7
TSM [22] (*i.e.* backbone only)	86.2	67.3	79.8	81.3
TSM_{HA} [17]	79.7	57.7	70.2	72.3
TSM_{EHF} (ours)	**89.2**	**69.2**	**80.5**	**82.5**
Reduced Resolution TSM_{EHF}	86.1	67.2	78.2	79.9
High Res Hands Only	87.6	67.1	78.6	80.4

Table 2: Comparison of model performance on both a segment level and on unseen extended assembly sequences utilising temporally aware label cleaning.

were informed to perform assemblies as normal, making no alterations for the camera. In total 38 full assemblies were recorded, corresponding to ∼6 hours of footage.

Labelling. The data is labelled using a fine-grained noun/verb labelling procedure with class selection guided by the standard operating procedure for the given assembly. We select fundamental interactions between the worker and either the product being assembled, a part/component or a tool - while ignoring unuseful or ambiguous actions for understanding the assembly process. In total seven primary interactions (verbs) were found: *Unbox, Pick Up, Put Down, Place, Tighten, Mark with Pen, Remove*. These interactions either occurred with a single object or with a primary and secondary object. In total 24 relevant actions verb/noun pairs were found between the operator and all objects within the assembly and are summarised in Table 1. To label the data we utilise a custom action segmentation labelling GUI, which we make public[2]. In total 186 632 from a total of ∼324 000 frames (∼57.6%) were found to contain an atomic action from our label list.

Properties. Our dataset follows a long tail distribution with the number of collected examples ranging from 38 (*Place Metal Bar* - only occurring once per assembly) to 610 (*Pick up Washer and/or Screw*) with average clip lengths ranging from as short as 0.96 seconds (*Put Down Spanner*) to 12.89 seconds (*Mark Screw(s) with Pen*). Total clip lengths of classes range from as low as 46 seconds (*Put Down Spanner*) to over 30 minutes (*Tightening Screw with Screwdriver*). While relatively modest in size our dataset is emblematic of many dataset collected within real world environments where the collection and labelling of large datasets is challenging due to quick turnaround

times and domain novelty. Compared to other action segmentation datasets ours is much more fine-grained than large datasets such as Breakfast [20] (77 hours - 10 actions) and larger in size than comparative fine-grained datasets such as GTEA [7] (28 minutes - 20 actions) and 50 salads [31] (4 hours - 17 actions). Our dataset is noticeably fine-grained as the background remains constant throughout all classes, with only subtle changes in spatial and temporal information between many classes. In addition, due to the noun/verb pairing process there are overlaps between some classes where the same noun (e.g tool or component) appears within multiple verb pairings (e.g screw or place) and vice versa. As such the ability to model both spatial and temporal information will be key to accurate classification.

4.2. Implementation Details

All models are trained within the Pytorch deep learning framework on segments extracted from 25 of the full assembly sequences, reserving the remaining 13 assembly sequences for unseen testing on both a segment and sequence level. See the supplementary information for a detailed account of parameter settings when training the models.

4.3. Evaluation Metrics

In order to test our models spatial and temporal reasoning we report results on both pre-cropped segments as well as temporal segmentation on unseen sequences. We report F_1 score on a segment level treating each pre-cropped segment as an single example regardless of length. When analysing segmentation results on an unseen sequence we report frame wise accuracy (Acc) as well as segmental edit distance, and segmental F1 score (as proposed by [21]) at various IoU thresholds (10%, 25% and 50%) to accurately account for over-segmentation errors.

4.4. Evaluation of High Resolution Hand Model

Table 2 compares the performance of our enhanced hand feature model (TSM_{EHF}) to an equivalent unenhanced TSM [22] model (both T=8 frames, with a temporal stride, τ=8) on a segment and extended sequence level. We further compare our model to an implementation of the simi-

---

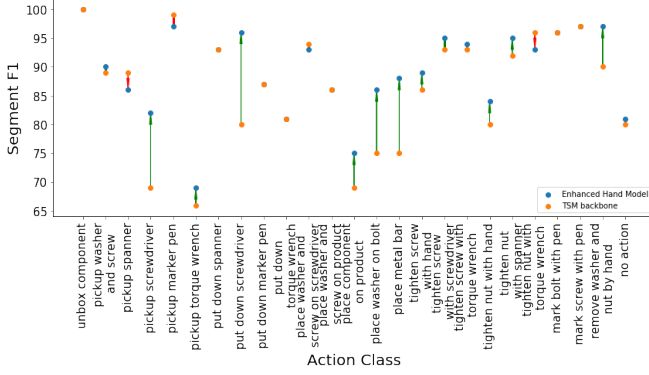[2]https://github.com/Matthewkm/Action-Segmentation-Labeller-GUI

Figure 3: Improvement in F1 score across classes on a segment level when comparing an 8 frame temporally aware TSM model to an equivalent model with our high resolution hand enhanced features.

Figure 4: Improvement in F_1@0.5 score across classes on a sequence level when comparing a single frame to an 8 frame temporally aware TSM model operating in a sliding window fashion.

lar hand attention (TSM_{HA}) work by Kobayashi *et al*.[3] [17] and a baseline single frame ResNet-50 model. We find our model outperforms the backbone TSM model and the hand attention model by Kobayashi.

Fig. 3 highlights how TSM_{EHF} improves the per class performance for nearly all classes compared to an equivalent backbone model, with the largest improvement seen in many classes with fine grained spatial differences, such as distinguishing between *place washer on bolt* and *place metal bar*. Our model does not harm the temporal reasoning as we see no decrease in temporally salient classes, and even see large improvements in *pick up/put down screwdriver* and *tighten nut with hand* vs *remove washer and nut by hand*, which require both fine-grained spatial and temporal reasoning. Only a small improvement is seen for *No Action*, suggesting our model is not improving the ability to distinguish between *No Action* and an action (as we see going from temporally unaware to temporally aware models - see section 4.5.1) but is extracting complementary enhanced spatial features, helping to distinguish between fine-grained spatial classes, without harming a models ability to temporally model frame wide features.

To further verify the performance improvements of TSM_{EHF} are due to high resolution hand data otherwise not available to the baseline TSM model, we implement two modified models also shown in Table 2. Firstly, just the high resolution hand section of our model (implementing the full ResNet-18 models with averaged softmax outputs) finding that the while the model performs better than the backbone TSM model on a temporally cropped segment level (despite having many more parameters) it performs worse on a sequence level, suggesting the larger frame wide features of the backbone model are useful for accurate segmentation under our framework (*i.e*. distinguishing boundaries of
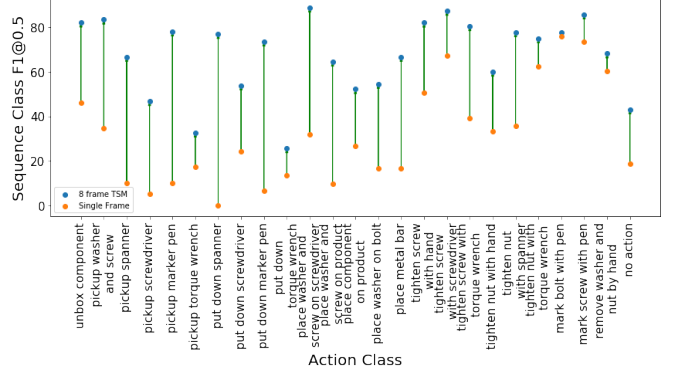
actions). Secondly, a low resolution hand version of our model, where hand input images are downsampled to match the resolution of the input image to the backbone model, essentially creating a model with the same number of parameters, but without increased resolution. As TSM_{EHF} outperforms both these models and the TSM baseline on a segment level we can see that high resolution hand features provide separate yet complementary features useful for classification, and that it is the access to higher resolution features that improves classification performance.

4.5. Real Time Action Segmentation Results

In the following section we compare the performance of temporally aware label cleaning as a method of deploying AR models to unseen extended sequences of assembly data. We show fine-grained temporally aware backbones are key to successful classification of short atomic actions and that temporally aware label cleaning is capable of matching the performance of encoder/decoder methods on temporally short atomic actions while maintaining the ability to operate in a near real time manner.

4.5.1 Importance of Short Range Temporal Learning

One key design choice within our model framework is that of the backbone model for classification and specifically how important is the ability for the backbone model to learn temporal features across multiple frames. Under our dense sampling strategy outlined in 3.5 there are two variables facilitating how much temporal information the backbone model can access from an input clip: the number of input frames, T, and the stride between them, τ. In order to investigate the importance of temporal reasoning we train a set of identical Temporal Segment Networks (TSN) with and without inserted TSM with varying numbers of input frames and temporal stride between input frames, with detailed results shown in section 4 of the supplementary material. Through our evaluation we find, for atomically short actions, that a temporally aware backbone operating on multiple frames is key to accurate classification on a sequence

---

[3]We note that Kobayashi *et al*. operate in an encoder/decoder structure, however, we re-implement their work following our method outlined in section 3.5 as well as utilise a temporally aware backbone structure to match our work.

Segmentation Approach	Model	Finetuned From	Real time	Acc	F1@{0.5,0.25,0.1}			Edit
Simple Sliding Window	TSN [39]	Kinetics	✓	74.1	24.9	34.6	39.5	31.2
	TSM [22]	Kinetics	✓	81.2	47.9	56.1	57.8	44.5
	TSM_{EHF} (ours)	Kinetics	✓	**81.8**	49.1	56.5	58.9	45.4
Temporally Aware Label Cleaning (ours)	TSN [39]	Kinetics	✓	73.6	49.1	67.8	72.9	65.4
	TSM [22]	Kinetics	✓	79.7	74.0	84.6	85.6	77.7
	TSM_{EHF} (ours)	Kinetics	✓	80.9	**75.8**	**85.6**	**87.2**	**79.9**
Encoder/Decoder [6]	ms-TCN	TSM Kinetics	✗	70.7	51.2	68.5	73.8	71.4
	ms-TCN	TSM Assembly Dataset	✗	**84.1**	77.6	85.3	86.3	79.6
	ms-TCN	TSM_{EHF} Assembly Dataset	✗	83.7	**78.4**	**87.0**	**88.4**	**83.6**

Table 3: Sequence Segmentation results for temporally aware label cleaning of sliding window predictions and ms-TCN using background emitted performance metrics as utilised in [6]. All models operating on T=8 frames with temporal stride τ=8

level. We find that performance saturates at T=8,τ=8 (real time FPS of 1.875 with the model only having access to 4.27 seconds of video) when temporally aware label cleaning is utilised.

Fig. 4 shows the improvement in individual class performance when changing from a single to an 8 frame temporally aware backbone model when performing action segmentationn in sliding window fashion. When utilising a single frame model many classes achieve very poor performances especially the more temporally salient classes, such as the *pick up/put down tool* and *place* classes, as a single frame model can't distinguish between picking up a tool and holding a tool (which falls under no action). As no action is a more prominent class, in the absence of the ability to distinguish between these actions and no action the model ends up classifying all these actions as no action. Once temporal reasoning has been enabled all classes are substantially improved, with some such as *put down spanner* seeing an improvement from 0 to 78 F_1@0.5. We also observe that the improvement in classification of temporally salient classes and no action is also much larger on a sequence level compared to a segment level (see supplementary information) suggesting a temporally aware backbone is key for accurate action segmentation when operating in a sliding window fashion, allowing the model to distinguish between actions and no action much more successfully.

4.5.2 Importance of Temporally Aware Label Cleaning

The first two sections of Table 3 show the performance before and after temporally aware label cleaning (section 3.5.2) of various models operating in a sliding window fashion on unseen assembly sequences from our fine-grained assembly dataset. Again we see that enhanced hand features continue to improve the model performance. Whilst we only see a modest change in frame level accuracy we observe that temporally aware label cleaning significantly improves the performance of our models on a sequence level, while allowing the models to maintain their near real time implementation[4]. This intuitively suggests temporally

---

[4]There is a small delays in model inference due to the need for Tτ/2 future frames for prediction and <Tτ/2 frames for cleaning - however this delay is <2 second for a 15 fps camera)

aware label cleaning is combating over-segmentation errors, which is further supported by the fact that as the short term temporal modelling ability of a model increase (*i.e.* TSN → TSM) the effects of the temporally aware label cleaning also increase, however much less of a change is observed for the enhanced spatial feature models (*i.e.* TSM → TSM_{EHF}).

4.5.3 Comparison to Encoder/Decoder Methods

Finally we compare our temporally aware label cleaning method to the commonly used encoder/decoder multi-sage Temporal Convolution Network (ms-TCN) [6]. Table 3 shows temporally aware label cleaning out performs a kinetics pretrained ms-TCN (utilising the same feature extraction model), while being able to operate in a near real time manner. We posit this is due to the fact that actions within our dataset are typically much shorter in nature than many other action segmentation datasets and do not necessarily occur in a fixed order, thus removing the advantage of extended temporal reasoning provided by an encoder/decoder method. We further show that features extracted from our assembly dataset trained models can be used to as base features to train ms-TCN models, providing significant improvements over the kinetics feature extraction (F_1@0.5 51.2 → 78.4) while reducing training time by ∼65% (200→70 epochs) showing that significant improvements can be obtained for other encoder/decoder methods via the use of in domain pretraining via the methods outlined in section 3.5.

5. Conclusion

In this paper we present a novel approach to utilising high resolution image features around an assemblers hand to improve action segmentation within real world assembly video. We show that through the use of short range temporally aware backbones, surround sampling and temporally aware label cleaning our model can be applied to extended unseen video sequences in a sliding window fashion, proving real time action segmentation required for many assembly HRC tasks. Further work will focus on applying models to other similar domain publicly available datasets as well as a more generalised network capable of high resolution features extraction from self determined areas of interest for use outside hand orientated datasets.

References

[1] Md. Al-Amin, Wenjin Tao, David Doell, Ravon Lingard, Zhaozheng Yin, Ming C. Leu, and Ruwen Qin. Action recognition in manufacturing assembly using multimodal sensor fusion. *Procedia Manufacturing*, 39:158–167, 2019. 25th International Conference on Production Research Manufacturing Innovation: Cyber Physical Manufacturing August 9-14, 2019 — Chicago, Illinois (USA).

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018.

[3] Jihoon Chung, Cheng hsin Wuu, Hsuan ru Yang, Yu-Wing Tai, and Chi-Keung Tang. Haa500: Human-centric atomic action dataset with curated videos, 2020.

[4] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, , Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision. *CoRR*, abs/2006.13256, 2020.

[5] Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description, 2014.

[6] Yazan Abu Farha and Juergen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation, 2019.

[7] Alireza Fathi, Ali Farhadi, and James M. Rehg. Understanding egocentric activities. In *2011 International Conference on Computer Vision*, pages 407–414, 2011.

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition, 2019.

[9] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition, 2016.

[10] Markus Funk, Thomas Kosch, and Albrecht Schmidt. Interactive worker assistance: Comparing the effects of in-situ projection, head-mounted displays, tablet, and paper instructions. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, page 934–939, New York, NY, USA, 2016. Association for Computing Machinery.

[11] Ankit Gupta, Dieter Fox, Brian Curless, and Michael Cohen. *DuploTrack: A Real-Time System for Authoring and Guiding Duplo Block Assembly*, page 389–402. Association for Computing Machinery, New York, NY, USA, 2012.

[12] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?, 2018.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[14] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3, 2019.

[15] Jonathan D. Jones, Cathryn Cortesa, Amy Shelton, Barbara Landau, Sanjeev Khudanpur, and Gregory D. Hager. Fine-grained activity recognition for assembly videos, 2020.

[16] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset, 2017.

[17] Takuya Kobayashi, Yoshimitsu Aoki, Shogo Shimizu, Katsuhiro Kusano, and Seiji Okumura. Fine-grained action recognition in assembly work scenes by drawing attention to the hands. In *2019 15th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, pages 440–446, 2019.

[18] Dan Kondratyuk, Liangzhe Yuan, Yandong Li, Li Zhang, Mingxing Tan, Matthew Brown, and Boqing Gong. Movinets: Mobile video networks for efficient video recognition, 2021.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[20] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 780–787, 2014.

[21] Colin Lea, Michael D. Flynn, Rene Vidal, Austin Reiter, and Gregory D. Hager. Temporal convolutional networks for action segmentation and detection, 2016.

[22] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[23] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2016.

[24] Liang Liu, Yong Liu, and Jiangning Zhang. Learning-based hand motion capture and understanding in assembly process. *IEEE Transactions on Industrial Electronics*, 66(12):9703–9712, 2019.

[25] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatiotemporal representation with pseudo-3d residual networks, 2017.

[26] Fabian Ranz, Titanilla Komenda, Gerhard Reisinger, Philipp Hold, Vera Hummel, and Wilfried Sihn. A morphology of human robot collaboration systems for industrial assembly. *Procedia CIRP*, 72:99–104, 2018. 51st CIRP Conference on Manufacturing Systems.

[27] Marcus Rohrbach, Anna Rohrbach, Michaela Regneri, Sikandar Amin, Mykhaylo Andriluka, Manfred Pinkal, and Bernt Schiele. Recognizing fine-grained and composite activities using hand-centric features and script data. *International Journal of Computer Vision*, 119(3):346–373, aug 2015.

[28] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. 2018.

[29] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding, 2016.

[30] Bharat Singh, Tim K. Marks, Michael Jones, Oncel Tuzel, and Ming Shao. A multi-stream bi-directional recurrent

neural network for fine-grained action detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1961–1970, 2016.

[31] Sebastian Stein and Stephen J. McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, page 729–738, New York, NY, USA, 2013. Association for Computing Machinery.

[32] Swathikiran Sudhakaran, Sergio Escalera, and Oswald Lanz. Gate-shift networks for video action recognition, 2019.

[33] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E. Shi. Human action recognition using factorized spatio-temporal convolutional networks, 2015.

[34] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014.

[35] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. 2019.

[36] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497, 2015.

[37] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition, 2017.

[38] L. Wang, R. Gao, J. Váncza, J. Krüger, X.V. Wang, S. Makris, and G. Chryssolouris. Symbiotic human-robot collaborative assembly. *CIRP Annals*, 68(2):701–726, 2019.

[39] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 20–36, Cham, 2016. Springer International Publishing.

[40] Peng Wang, Hongyi Liu, Lihui Wang, and Robert X. Gao. Deep learning-based human motion recognition for predictive context-aware human-robot collaboration. *CIRP Annals*, 67(1):17–20, 2018.

[41] Xi Vincent Wang, Zsolt Kemény, József Váncza, and Lihui Wang. Human–robot collaborative assembly in cyber-physical production: Classification framework and implementation. *CIRP Annals*, 66(1):5–8, 2017.

[42] Qianqian Xiong, Jianjing Zhang, Peng Wang, Dongdong Liu, and Robert X. Gao. Transferable two-stream convolutional neural network for human action recognition. *Journal of Manufacturing Systems*, 56:605–614, 2020.

[43] Ceyuan Yang, Yinghao Xu, Jianping Shi, Bo Dai, and Bolei Zhou. Temporal pyramid network for action recognition, 2020.

[44] Jianjing Zhang, Peng Wang, and Robert X. Gao. Hybrid machine learning for human action recognition and prediction in assembly. *Robotics and Computer-Integrated Manufacturing*, 72:102184, 2021.