ISSN 2464-4617 (print)
ISSN 2464-4625 (online)
Computer Science Research Notes - CSRN 3301
http://www.wscg.eu
WSCG 2023 Proceedings

Semi-Supervised Learning Approach for Fine Grained Human Hand Action Recognition in Industrial Assembly

Fabian Sturm
Bosch Rexroth AG
Lise-Meitner-Strasse 4
89081 Ulm, Germany
fabian.sturm@bosch.com

Rahul Sathiyababu
Bosch Rexroth AG
Lise-Meitner-Strasse 4
89081 Ulm, Germany

Elke Hergenroether
University of Applied Sciences Darmstadt
Schoefferstrasse 3
64295 Darmstadt, Germany

Melanie Siegel
University of Applied Sciences Darmstadt
Schoefferstrasse 3
64295 Darmstadt, Germany

Abstract

Until now, it has been impossible to imagine industrial manual assembly without humans due to their flexibility and adaptability. But the assembly process does not always benefit from human intervention. The error-proneness of the assembler due to disturbance, distraction or inattention requires intelligent support of the employee and is ideally suited for deep learning approaches because of the permanently occurring and repetitive data patterns. However, there is the problem that the labels of the data are not always sufficiently available. In this work, a spatio-temporal transformer model approach is used to address the circumstances of few labels in an industrial setting. A pseudo-labeling method from the field of semi-supervised transfer learning is applied for model training, and the entire architecture is adapted to the fine-grained recognition of human hand actions in assembly. This implementation significantly improves the generalization of the model during the training process over different variations of strong and weak classes from the ground truth and proves that it is possible to work with deep learning technologies in an industrial setting, even with few labels. In addition to the main goal of improving the generalization capabilities of the model by using less data during training and exploring different variations of appropriate ground truth and new classes, the recognition capabilities of the model are improved by adding convolution to the temporal embedding layer, which increases the test accuracy by over 5% compared to a similar predecessor model.

Keywords

Human Action Recognition, Industrial Assembly, Semi-Supervised Learning, Transfer Learning, Transformer

1 INTRODUCTION

The full automation of production processes in industrial production lines has shown that the contribution of humans cannot be completely replaced by machines, yet. This adapted attitude toward full automation is primarily due to monetary reasons, such as increased fixed costs due to the maintenance of the machines used in the process, but also to reasons such as lower social acceptance due to the elimination of certain occupational groups. The advantages resulting from and utilized by humans compared to today's machines are primarily evident in manual assembly work. Humans are able to process a high product variance, to adapt to new or optimized processes without additional technical effort, and to recognize and adjust to unexpected problems and assemble components very accurately and precisely. However, the increasing variance of products and shorter time-to-market for companies is becoming a disadvantage for people due to the increase in workload. This workload leads to a higher error rate, especially when it comes to assembling products. The main reason for this is lack of concentration, especially due to long work shifts, as well as inattention or distraction. In addition, the assembly worker needs more and more in-depth knowledge to assemble the products correctly and has less time for training and instruction. These circumstances lead to a lack of expertise and reduce positive human flexibility. The errors in assembly, which initially go undetected, then continue through the entire production process until they are noticed during quality control or, in the worst case, during operation of the product. The consequences are inconvenience to production, an increased number of defective products or damage to the company's image because of the low quality. One possible solution to avoid the aforementioned problems in assembly without affecting the economic parameters are assistance systems for the worker.

These assistance systems guide the employee through the assembly process and point out errors. In addition to detecting the objects used, such as tools and components, the first step in this observation of assembly tasks is to observe the fine-grained human actions performed by the hands of the assemblers. For the recognition and description of these fine-granular hand actions, a stacked deep-learning architecture is used. This deep-learning model architecture consists of an existing feature extractor for hand detection and tracking and subsequent classification of the sequences by an existing for this specific use case adapted spatio-temporal transformer model based on the findings of [Liu+21]. Beside the presentation of the architecture and usage of an available dataset for fine-grained human hand actions in industrial environment, [Stu+23] the main focus of this paper is the semi-supervised model training procedure dealing with the general problem in an industrial environment with lots of data but few labels. This circumstance is a relevant point at the latest when the trained basic application needs to be adapted to a new real world use case in order to check the scalability of the model. Especially in an industrial environment, this must be done as quickly and cost-effectively as possible and, without time-consuming labelling effort, as is the case with traditional deep learning methods. In this work, pseudo-labelling transfer learning experiments are therefore investigated to improve the stability of each class and the overall performance and generalization possibilities of the model. A base model trained supervised on different combinations of assembly tasks with the highest F1-Score[GBV20] from the "Industrial Hand Action Dataset V1"[Stu+23] is pretrained to provide these weights as initialization for an industrial real world use case from a laboratory environment. For the subsequent real world use case, it is assumed that partial labels based on the job description and example assembly steps for training new employees exist, which are in this case a combination of novel classes from the "Industrial Hand Action Dataset V1". The pretrained weights are transferred to a new classifier architecture, which is subsequently fine-tuned on the novel classes in a semi-supervised procedure by using at first the labeled example data from the customer and initialize the model for the training on unlabeled data. Afterward, the pseudo-labeling approach follows. This procedure shows how this method of semi-supervised training can positively affect the classification results in an industrial environment. It provides more stability and generalizability compared to supervised approaches for an industrial real world use case and is based on a spatio-temporal transformer network for fine-grained human hand actions. In the following Sections, the training data set is introduced in Section 2 followed by the stacked model architecture in Section 3 existing of the hand detector in Section 3.1 and the spatio-temporal transformer model for the classification task in Section 3.2. The related work 4 of pseudo-labeling a method of semi-supervised learning is presented in Section 4.1 and the shown approaches are revisited under consideration of the usability in this specific industrial human hand action recognition approach in Section 4.2, before the experiments are examined in Section 5 and finally evaluated, compared and concluded in Section 6 and Section 7.

2 INDUSTRIAL DATASET

The "Industrial Hand Action Dataset V1", a vision based industrial hand assembly dataset introduced in [Stu+23] consists of 12 fine-grained hand action classes for industrial assembly. With 459,180 frames in the basic version and 2,295,900 frames after spatial augmentation, uneven distributed it belongs to one of the larger datasets. It is the first dataset of its kind to tackle the real world issues which occur in an industrial environment. Compared to other freely available datasets tested in [Stu+23], it has an above-average duration and, in addition, meets the technical and legal requirements for industrial assembly lines. Furthermore, the dataset contains occlusions, hand-object interaction, and various fine-grained human hand actions for industrial assembly tasks that were not found in combination in similar examined datasets [Stu+23]. The recorded ground truth assembly classes are selected after extensive observation of real-world use cases. The usability of the dataset for training sequential deep learning models was confirmed in [Stu+23] with a test accuracy of 86.25% before hyperparameter tuning. The architecture is based on an adapted version of the gated transformer network model of [Liu+21], presented in more detail in section 3.2.

3 MODEL ARCHITECTURE

The processing of full image sequences is a very resource intensive task when it comes to deep learning model training. Since the focus of the model architecture in the described industrial use case is on the hands, the decision was made to apply a skeleton-based human hand action recognition approach. Hand coordinates are extracted from frames to get a reduced feature space data sequences. This lowers the computational effort for the subsequent classification task by providing to the model 2.5 dimensional coordinates per frame without any noise from tools, unnecessary objects or other assemblers. This can be seen in Figure 2, or for the classification of unnecessary information of the in industrial environment usual static background, see Figure 1. Subsequently, for the classification of the performed assembly task of the hands, the sequences of coordinates per frame are provided to the adapted Gated Transformer Network by [Liu+21]. They achieved good re-

ISSN 2464-4617 (print)
ISSN 2464-4625 (online)
Computer Science Research Notes - CSRN 3301
http://www.wscg.eu
WSCG 2023 Proceedings

sults on the initial work on the dataset in [Stu+23] and on comparable datasets in [Liu+21].

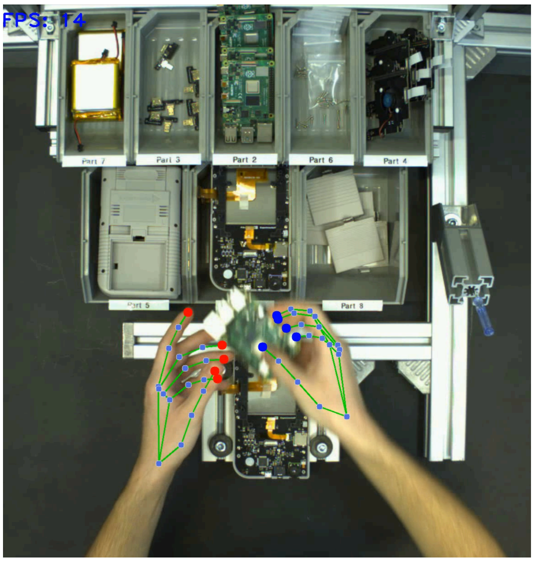

Figure 1: Static Background in Industrial Environment

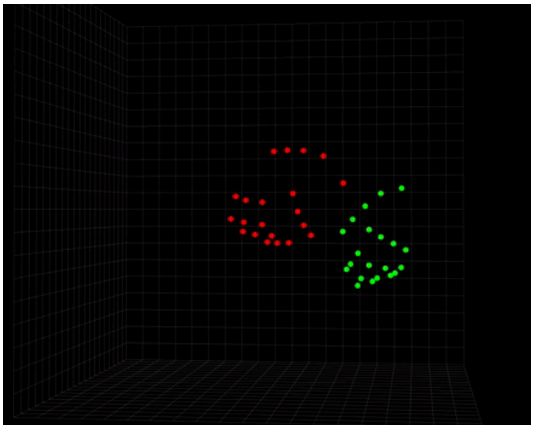

Figure 2: Feature Space for Sequential Model Training

3.1 Hand Keypoint Extractor

The keypoint extraction is based on an approach similar to the architecture of Google's MediaPipe Hands solution, with a palm detection model and a hand feature detector on top [Zha+20a]. The output of this model architecture is able to provide 21 2.5D coordinates for different landmarks, which are used as pre-extracted features or keypoints for the subsequent transformer model architecture. The keypoint extractor starts with a palm detector network, which is a single shot detector optimized for mobile real-time usage [Zha+20a]. It takes advantage of first recognizing the palms through a bounding box, which is a more stable approach for the model than first recognizing the entire hands with fingers due to its square shape. Further, an encoder-decoder feature extractor similar to a Feature Pyramid Network [Lin+16] is used to detect the palm across a large range of scales. Focal Loss [Lin+17] is used as the loss function, since it handles the imbalance between background segment detections and actual palm detections better by reducing the influence of background detections during training [Zha+20a]. Once a palm has been detected, a cropped image is generated based on the mentioned detection. This includes more image data than the palm bounding box itself in order to contain the entire hand. This cropped image is then provided to a second network consisting of a convolutional neural network to detect the hand landmarks and outputs the 21 hand landmarks with 2.5D coordinates. The values consist of x, y values which are calculated by the size of the frame and a depth value relative to the wrist landmark. Furthermore, a probability of a hand being present and the handedness is provided. Based on the detected landmarks, a new crop area is calculated to try to keep the hand within this area. The next image in the sequence is then cropped to this new value straight away, without the single-shot detector running again. Only if the probability of a hand being present is below a certain threshold, the single-shot detector runs again on the entire image [Zha+20a]. Since the convolutional neural network has to run only on a smaller cropped image and the single-shot detector only has to scan the entire image if a hand has been lost, the number of compute cycles required during inferencing is reduced.

3.2 ConvGTN

The output of the hand keypoint extractor leads to the second part of the architecture. This part classifies the temporal and sequential correlation of the previously extracted time series of keypoints by a gated transformer network from [Liu+21] which is adapted to this use case, see Figure 3. [Liu+21] achieved state-of-the-art results on 13 multivariate time series classification tasks in the domain of Natural Language Processing (NLP), but also human action recognition tasks which are comparable to this use case[Liu+21]. The architecture is based on a two-tower transformer, where the encoder in each tower capture time-step-wise and spatial-channel-wise attention. To merge the encoded feature of the two towers, a learnable weighted concatenation is used as a gate before the final fully connected layers. This gate decides which tower of the network provides more important features for the final classification during backpropagation. For the improvement of the prediction results of the model, an additional Conv1D[1] with kernel size of 5 is implemented. This additional convolution helps the model to find better correlation between the hand keypoints [21*3*2] [keypoints per hand*xyz coordinates*hands], in the temporal embedding of the model and leads to better gradients

[1] https://pytorch.org/docs/stable/generated/torch.nn.Conv1d.html

in the temporal tower at each time step of the input data, [8,126,100] [batch size, features, sequence length]. The convolutional layer is followed by a linear layer with 512 input and 512 output features and leaky rectified linear unit (LeakyRelu) as activation function [Xu+15], which is added to the temporal embedding layer for better generalization performance, see Figure 3.

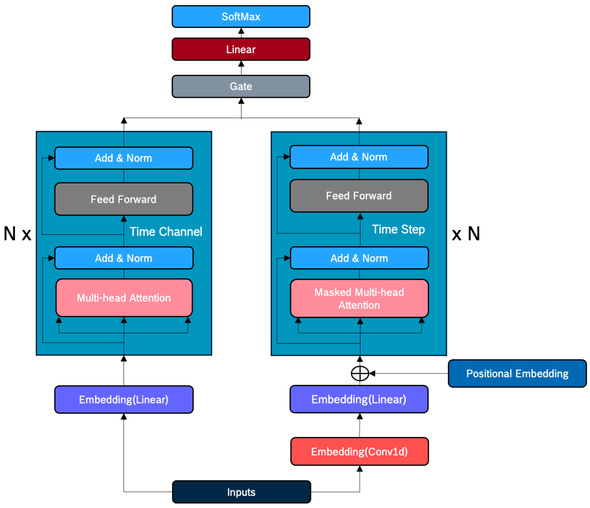

Figure 3: ConvGTN Model Architecture

4 RELATED WORK

The choice of a semi-supervised training method for the presented model architecture is based on the large amount of recurring patterns in the data generated by industrial processes and the precise instructions given to assemblers before starting an assembly task. In the process, the employee is first shown assembly tasks as ground truth via training videos, which he or she is then expected to perform. When performing the assembly steps, a wide variety of tasks occur, especially by different employees. In addition, there are also tasks that are similar but must be performed differently due to the variance of the components. However, these have the same specifications in the task description and can thus cause a high variance and positive influence on the generalizability of the model to be trained. The approach envisaged for this is to train a ground truth on a common specification and use it as a basis to transfer, recognize, and learn the recognition of new or similar assembly steps faster. For this purpose, a base model is first trained on labeled data, and this knowledge is then used as base initialization knowledge for new classes. This is done by replacing and re-training the final classification layers trough fine-tuning [Far+21]. In order to remain appropriate to the real condition in the production, only a small portion of marked data is initially used. The unlabeled new data is then iteratively labeled by the model, and the entire model is re-trained from the transferred state to fine tune parts of the model on the new knowledge. For this purpose, a method of semi-supervised self-training [GB04] is used, more specifically pseudo-labeling. The labeled data set is augmented by self-training to detect better relationships in the data through self-supervised learning in the unsupervised domain [Zho+18].

4.1 Self-Training for Classification Tasks

Most of the work in the subcategory of self-training, pseudo-labeling, concentrate on image classification [Yan+21][Wen+21]. Especially in this use case of fine-grained human hand action recognition or skeleton based human action, very few studies exists whether it is based on videos nor previously extracted features [Xu+21][ITP14][Xia+21]. The difference between a training approach with labels and without labels like in this case is that the algorithm uses the model's own trusted predictions to create the pseudo-labels for unlabeled data and can add more training data by using existing optimally self-labeled data to predict the labels of the unlabeled data [Ami+22]. The firstly proposed approach of pseudo-labeling by [Lee13] uses a small set of labeled data to iteratively train a model in combination with a large set of unlabeled data. A small set of labeled data is used to iteratively train a model in combination with a large set of unlabeled data. Using cross entropy loss, this involves training a base model, which is then used to make a prediction on a batch of unlabeled data. These predicted labels are then added to the labeled data set, the model is trained again on the data set, and the next batches are predicted until, in the optimal case, there is no unlabeled data left. The loss function is defined as follows, see Equation 1 [Lee13].

$$L = \frac{1}{n}\sum_{m=1}^{n}\sum_{i=1}^{C} L(y_i^m, f_i^m) + \alpha(t)\frac{1}{n'}\sum_{m=1}^{n'}\sum_{i=1}^{C} L(y'^m_i, f'^m_i) \tag{1}$$

In this equation, n defines the number of mini-batches in labeled data for stochastic gradient descent (SGD) and n' the number of mini-batches for unlabeled data. f_i^m is the output unit of m samples in labeled data, and y_i^m is the label of the samples. f'^m_i is a sample as well, just for unlabeled data, and y'^m_i is the pseudo-label of that. $\alpha(t)$ is a coefficient balancing the supervised and unsupervised loss terms by increasing the influence of the unsupervised loss per each iteration. In general, the pseudo-label approach takes entropy minimization to get the pseudo labels with the highest confidence as the ground truth for unlabeled data [Lee13]. An improved version of the traditional pseudo-labeling is the Meta Pseudo-Labeling (MPL) [Pha+20]. In this approach, a teacher model assigns distributions to the inputs and thus trains the student model. While training the student, the teacher observes the student's performance on a validation set and learns to generate target distributions so that when the student learns from such distributions, the student performs well in validation. The MPL training procedure consists of two alternating processes. The teacher generates the conditional class distribution for the student's training. This pair is then fed into the student network to update its parameters based on the cross entropy loss. After the student network updates its parameters, the model evaluates the new parameters based on the samples from the validation dataset. Since the student's new parameters depend on the teacher, the gradient of loss can be calculated based on the dependence to update the teacher's parameters. Similar to this two model approach is Cross-Model Pseudo-Labeling, which focuses on the problem of having less labeled data where a single model is not able to provide good enough pseudo labels [Xu+21]. As shown in their experiments, a deeper model can find better spatial correlations, while a smaller model is better at detecting temporal correlations sequentially. This leads to the result that two models that are different sizes can be used for pseudo-labeling to better distinguish their found labels. The models predict each other's pseudo-labels and use them to train or, more precisely, subsequently back propagate the unsupervised loss through themselves to improve performance. [Xia+21] made the same results for spatial and temporal correlations, but in video-based action recognition, especially in cases with fewer data. They improved performance with a blockwise dense alignment strategy and cross modal contrastive learning, focusing the model on the temporal dynamics of videos by computing a temporal gradient. These methodologies shown so far are used in many approaches to pseudo-labeling and require that the training parameters of the model be set to optimal values to achieve optimal performance. The problem with these approaches is that there must be confidence in the model to successfully generate pseudo-labels. This means that the model must generate the correct labels, otherwise incorrect labels are generated, the model subsequently learns these self-generated incorrect labels, and predicts incorrectly again due to worse overall performance by iteratively adding noise to itself to the training process. To avoid these problems, [Ber+19] uses a method called MixUp which guesses low-entropy labels for augmenting unlabeled examples in each batch, and then mixes augmented labeled and unlabeled data together by using traditional regularization methods. The augmentation is made by consistency regularization within the unlabeled and labeled data, and afterward the cross entropy loss is used to minimize the loss between the guessed labels and the unlabeled data. FixMatch, a state-of-the-art method, uses a combination of consistency regularization and pseudo-labeling, by requiring that the predictions of strongly augmented data can be paired with the predictions of weakly augmented data to create a labeled sample [Soh+20]. Beside this approach to prevent wrong labels by augmenting and compare the data, which is in this case not possible since the data is unknown, the wrong label prediction can be avoided by adding something similar to a threshold like in the case of curriculum labeling [Cas+20]. In this approach, self-paced curriculum principles are applied and additionally, to avoid concept drift [Lu+20], the model parameters before each pseudo-labeling cycle are reinitialized from scratch. [Cas+20] uses successful curriculum learning approaches from [Ben+09], where a model is first trained with simple examples and then iteratively progresses to more difficult examples. As described, the difficulty is to create a curriculum that goes not too fast but also not too slow over the initially simple examples. To successfully learn the general features and to iteratively store the knowledge of the weights, a percentile is used. [Ber+22] uses a similar curriculum approaches for pseudo-labeling of NLP tasks by tracking the generalization and overfitting progress. Regarding their findings, especially in fewer data cases, pseudo-labeling is not successful because of overfitting in an early stage of pretraining with labeled data. The recommendation is to start the semi-supervised training very early in the training process by dynamically controlling the pseudo-labels with curriculum to avoid overfitting and stabilize the model.

4.2 Revisiting Self-Training Methods

Several approaches in the semi-supervised self-training domain of pseudo-labeling exist [Ami+22]. However, many are based on a prior spatial augmentation approach, which is relatively easy to implement when images are available [Soh+20][Ber+19][Wen+21][Yan+21]. Since in this showed use case for human hand action recognition in industrial assembly lines, where the features to be used have already been extracted, this is not directly possible at the image level. Furthermore, after the first feature extraction it is necessary to maintain the structure of the hand keypoints, which is why the individual keypoints may not be augmented without further ado. This shows that there is little experience in the described methods that focus on this particular method for an industrial use case and employ it in combination with a transformer encoder model like the ConvGTN from Section 3. [Pha+20], [Xu+21] and [Xia+21] show that it helps to work with separate models on the spatial and temporal part with different focus like it is already in this model structure and seems like a promising approach. But with this teacher student approach one needs to keep track of the confirmation bias which restricts the performance of the student by the teaching performance of the teacher [Ara+19]. Moreover, especially in the approach of this work, no standard pseudo-labeling method can be applied, because due to the many parameters, of the ConvGTN, in this case over 18M., it cannot be assumed that the correct settings of the hyperparameters can be applied at the training. [Cas+20] and [Ber+22] proved that it helps to converge in the semi-supervised search space faster by using a threshold as well as curriculum and percentile. The focus of the experiments on these methods was based on the described approaches to self-training for fine-grained human hand action recognition in industrial assembly.

5 SELF-TRAINING EXPERIMENTS

Data split for Experiments

For the experiments, a transfer pseudo-labeling approach with curriculum is implemented and compared with several combinations of base and novel data, always under consideration of 7 base and 4 novel class data distribution.[2] The decision for this partitioning is made to have a strong pretrained model, for weight initialization of the base classes under consideration of just a few examples of novel classes as it is recommended by [Zha+20b]. Besides, it was recognized, if the novel classes are reduced by one class, not enough data is provided for the semi-supervised training and vice versa for a base model pretraining with good performance. Since there are 35 different distribution possibilities for the 7/4 split, five distributions were selected from [Stu+23] depending on the F1 score of the ground truth results of the supervised training on all 11 classes. The selection was based on the best matching classes to these specific industrial conditions, see table 1. The ground truth supervised model training with the new created class `Assembly_Step7` is added as experiment 1 for comparison.

Supervised Pretraining & Weight Transfer

After the data split, a full ConvGTN model from Section 3.2 is supervised pretrained on several combinations of 7 base classes with a seed of 42, a batch size of 6 and a learning rate of 1e-4. These model weights are transferred supervised to 10% of the 4 novel classes by freezing the whole model with all layers of the encoder but the feedforward of the encoders and the final classifier after the gate, see for comparison Figure 3. The learning rate in the transfer process is set per layer in the Adam optimizer to 1e-3 in the encoder, 1e-4 in the classifier and is tracked for minimization by a `ReduceLRonPlateau`[3] scheduler which tracks the learning rate by patience of 7, factor of 0.1 and minimal decay of learning rate of 1e-9. These new weight initialization of the feed forward layers of the encoder and the classifier helps afterward the curriculum pseudo-labeling approach to converge faster and was trained in 40 epochs by a batch size of 8.

Pseudo-labeling Based Self-Training

The difference in the semi-supervised self-training to the initial setting is that only one of four layers of the encoder of the model is frozen completely. A similar approach as a method that applies higher learning rates to top layers and lower learning rates to bottom layers is used. This procedure took place since existing experiments using similar encoder architectures like BERT [Dev+18] showed that using the complete network for transfer was not always the most effective choice, because transferring the top pre-trained layers can slow down learning and decrease the performance [Zha+20b]. This can be explained as different layers in transformer structures usually capture different kinds of information. Bottom layers often encode more common, general, and broad-based information, while the top layer closer to the output encodes information more localized and specific to the task on hand [Zha+20b]. This procedure is partly accomplished by setting manually the learning rate of the top layer and using a multiplicative decay rate to decrease the learning rate layer-by-layer from top to bottom [HR18]. For these experiments, the learning rate per layer was set to 1e-3 in the embedding layers, 1e-2 in the gate, and 1e-3 in the final classifier, see Figure 3 for comparison of the layers. The encoder part was trained further than the other layers during pre-training and fine-tuning, therefore the learning rate was set to 0.5e-3. In contrast, to the recommendation to use SGD for semi-supervised learning approaches, Adam optimizer is used to adapt the optimizer to the transformer architecture as it has proven to be a successful optimizer approach in attention models [Zha+19]. Additionally, L2 regularization is used to avoid that the pre-trained target weights deviate too much from the initial weights. For the same reason, the scheduler patience was reduced to 5. After the pre-trained weight transfer to the 10% novel classes, a first initial iteration is used to predict pseudo-labels on the unlabeled dataset. Therefore, a bottom to top approach is used. First, all the unlabeled data is shown to the model with low threshold more precise a percentile, which is computed in a 0.2 step to find widespread patterns over all the data in 150 epochs per iteration.

[2] During preprocessing, a too similar distribution between `Assembly_Step7` and `Assembly_Step8` was detected, which was caused by the augmentation of the augmented standard dataset and could therefore not be clearly separated from each other. Therefore, `Assembly_Step7` and `Assembly_Step8` were merged to one class, `Assembly_Step7`. The now appearing majority of the class compared to the other classes was compensated afterward by weight balancing and by combining only each second example of both classes to `Assembly_Step7`.

[3] `\https://pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html`

After an iteration the predicted labels are added to the dataset, removed from the unlabeled dataset and the model weights are reset to the initial weights from the fine-tuning with again a completely new classifier. The following iterations are done by freezing the convolutional layer and both embedding layers in the first transformer layer, and unfreeze and train all the other layers that are left, see Figure 3 [LTL19]. This procedure is repeated until the percentile reached 100% and the model is confident to predict the right labels to the data. For the final validation, the best model of the last iteration is used because this model had access to most of the data. The complete self-training architecture with curriculum pseudo-labeling can be seen in Figure 4.

5.1 Experimental Setup

The improved ground truth results of the ConvGTN from Section 3.2 are taken as a base for these experiments, especially the test accuracy and F1-Score per class is therefore under deeper consideration. The 7/4 data split was done as shown in Table 1. Additionally,

Exp.	Class Split	7 Base Assembly_Step	4 Novel Assembly_Step
1	0Best & 4Worst	1,2,3,4,5,7,12	6,9,10,11
2	1Best & 3Worst	1,2,3,4,5,7,9	6,10,11,12
3	2Best & 2Worst	2,3,4,5,7,9,10	1,6,11,12
4	3Best & 1Worst	2,3,4,6,7,9,10	1,5,11,12
5	4Best & 0Worst	3,4,6,7,9,10,11	1,2,5,12

Table 1: Dataset Split for Experiments

to the F1-Score comes the observation which combination of base and novel assembly tasks from [Stu+23] works best for the semi-supervised fine-tuning with curriculum learning approach. It is also assumed that not all classes can provide unique features for generalization and lead to good performance of the model when generalizability is considered.

5.2 Model Training Environment

The model architecture was created in PyTorch and stacked on top of Googles framework MediaPipe Hands. The training and hyperparameter tuning was done in Microsoft Azure on a STANDARD_NC6 with 6 vCPUs and 56 GiB Memory. The final model training was done on a GPU which corresponds to half a K80 card with 12 GiB, and a maximum of 24 data disks and 1 NCiS in a duration of 2 hours and 24 minutes up to 4 hours for the pretrained model and 4 hours and 5 minutes up to 6 hours for the self-trained model both depending on the split of the classes.

6 COMPARISON & RESULTS

For a better comparison of the overall results in Table 3, the ground truth F1-Score of each class by training supervised on 100% of the data is provided in Table 2. With this initial supervised training method, the ConvGTN reached an overall ground truth test accuracy of 91.18% on the "Industrial Hand Action Dataset V1" [Stu+23]. This result is 5% higher as the test accuracy without the convolutional layer in [Stu+23]. For the following experiments, these target classes of the dataset are split, into 7 base classes for pretraining and 4 novel classes for the downstream tasks. It can also be assumed from previous literature reviews in Section 4 that a strong base model has a positive effect during training on novel models. The split into base and novel classes depends on the results of the F1 score per class from table 2 by dividing the classes into high (best) and low (worst) F1-scores for the novel downstream task. For the exact class split per experiment, see Figure 1.The final results per experiment are presented in table 3 showing each data split per experiment by column and within each column always 3 different training runs with the F1 score results per sub-column. Training on only 10% labeled data is presented in sub-column "10% Sup" for the comparison that semi-supervised training helps. Sub-column "PreTrained + 10% Sup" showing that pre-trained weights help fine-tune to 10% of new data, and the final target, pre-training on baseline data and fine-tuning to 10% of new data as initial weights, followed by 90% curriculum learning in the sub-column "PreTrained + 10% Sup + 90% SemiSup". As additional information in the last row of the table, the overall test accuracy is presented to compare the overall performance over the different combinations of classes depending on best to worst F1-scores of the novel classes after each training experiment. The goal over all experiments is to reach nearly the same F1-Score with the semi-supervised approach, as with 100% labeled data. These results help to see how the pseudo-labeling improves the scores in the experiments. Additionally, only the F1-Scores of the novel classes of each of the experiments are added to the split classes, since only there an improvement helps for making a final result for providing deeper insides into the proof of scalability in deep learning models in industrial environments.

Effect of Pretraining

Compared to the supervised training on only 10% of labeled data, the additional pretraining of the model on the base classes shows as expected in Table 3 an improvement in the overall test accuracy in all variations of the novel classes and experiments. It is also visible, that with a better F1-Score in the novel class, not only the "Test Acc." improves but also the F1-Score in each experiment. Only some small outliers are visible in `Assembly_Step12`, with a reduction from 0.91 to 0.89 in the second experiment and from 1.00 to 0.80 in the fifth experiment, which can in both cases be traced back to bad features in base and novel training. These bad features could lead to negative transfer learning,

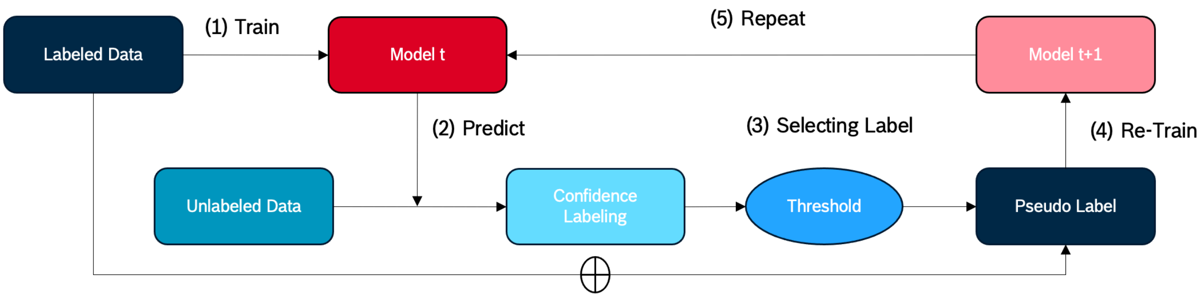

Figure 4: Self-Training Architecture

Assembly_Step	1	2	3	4	5	6	7	9	10	11	12	Total Acc.
F1-Score	0.94	0.93	0.91	0.92	0.93	0.87	0.92	0.91	0.90	0.86	1.00	91.18%

Table 2: Ground Truth Results

like it is experienced beside these experiments in [Wan+18], [PY10] and [Ros+05]. This behavior during model training needs to be examined deeper in future experiments. Nevertheless, the result can be made as expected that robustness of a pre-trained model and amount of data that is used for fine-tuning matters for the overall model performances.

Effect of Pretraining with Curriculum Self-Training By adding the semi-supervised curriculum learning approach with 90% of unlabeled data, the experiments showed that this procedure gives nearly all models an increase in "Test Acc." by almost reaching the ground truth test acc. over all classes in the last experiment with 86.07% compared to the initial ground truth of 91.18%. The overall test accuracy of different variations of novel classes trained with 90% unlabeled data in the self-training method is always higher than the training on 10% labeled data and adding pretrained model results. Since more data helps in fine-tuning and semi-supervised learning approaches beside the increase of the number of best F1-Score classes in the novel classes. In experiment 5 of the self-training method, the performance increased by 18.32% from 67.75% to 86.07% when compared to experiment 3 with the 4 worst novel classes. But even in these experiments with more data, negative transfer learning is visible. Especially in the cases of experiment 3 and 5 where weak F1-Score classes are added to the novel classes. This is especially the case for class `Assembly_Step6` where the F1-Score dropped from a 0.55 to 0.54 score and from 0.78 to 0.75 score which also affects the final test acc. with 1.4% less. Since this behavior has happened in experiment 5 in 3 out of 4 experiments, the negative transfer learning needs to be further evaluated in the semi-supervised learning approach as well. Additionally, it can not be excluded that the combination of 2 best and 2 worst novel classes leads to bad results in model training because of the possible close relation and similar movements in the actions of the classes.

7 CONCLUSION & OUTLOOK

This approach shows, first, that adding a convolutional layer in the investigation by a spatial tower improves the performance of a spatio-temporal transformer model. Secondly, the main findings in this work are that fine-grained human hand action recognition on a limited amount of novel data can indeed be improved by pre-training of a base model and subsequent usage of a self-training approach based on curriculum labeling to raise the final evaluation results and generalization possibilities of the model. Therefore, it is also important which classes are part of the base model training and how strong and obvious novel classes need to be for the model. Which means having a strong pretrained model helps to improve the results, but also strong novel classes can help if enough data is available. It was also shown that a stable model can be trained even with a small amount of labeled data. This confirms that industrial environments are ideal for scaling deep learning approaches and gives deeper insights for the creation of a pretrained model to prove the scalability in industrial environment. Besides, it shows how the fine-tuning in transformer models needs to happen by freezing only specific layers of the transformer architecture but also use different learning rates per layers. In addition, compared to the traditional methods of using an SGD optimizer in semi-supervised training, Adam was used. Since a lot of human fine-grained hand actions look similar, negative transfer learning was experienced probably because of the similar movements from base to novel classes which will be further evaluated in the following experiments, by investigating approaches to prevent negative learning and to improve the generalizability of the model, by self-attention approaches for the recognition of fine-grained human hand actions in industrial assembly.

REFERENCES

[Liu+21] Minghao Liu, Shengqi Ren, Siyuan Ma, Jiahui Jiao, Yizhou Chen, Zhiguang

10% Sup | PreTrained + 10% Sup | PreTrained + 10% Sup + 90% SemiSup					
Class\Experiment	0Best & 4Worst	1Best & 3Worst	2Best & 2Worst	3Best & 1Worst	4Best & 0Worst
Assembly_Step1			0.22 | 0.67 | 0.74	0.40 | 0.74 | 0.69	0.43 | 0.57 | 0.85
Assembly_Step2					0.42 | 0.70 | 0.88
Assembly_Step5				0.31 | 0.50 | 0.88	0.22 | 0.70 | 0.81
Assembly_Step6	0.25 | 0.55 | 0.54	0.44 | 0.57 | 0.69	0.52 | 0.78 | 0.75		
Assembly_Step9	0.67 | 0.80 | 0.76				
Assembly_Step10	0.46 | 0.53 | 0.59	0.50 | 0.80 | 0.72			
Assembly_Step11	0.30 | 0.47 | 0.60	0.53 | 0.70 | 0.73	0.57 | 0.74 | 0.71	0.14 | 0.71 | 0.71	
Assembly_Step12		0.91 | 0.89 | 0.95	0.83 | 0.91 | 0.89	0.50 | 1.00 | 0.85	1.00 | 0.80 | 0.92
Test Acc. in %	40.54 | 59.46 | 63.82	56.67 | 72.41 | 74.81	54.84 | 76.67 | 75.27	32.41 | 71.43 | 77.22	41.94 | 67.75 | 86.07

Table 3: Experimental Scores per Class on Pretrained Model on 7 Base Classes + Semi-Supervised 4 Novel Classes

Wang, and Wei Song. "Gated Transformer Networks for Multivariate Time Series Classification". In: *CoRR* abs/2103.14438 (2021). arXiv: 2103.14438. URL: https://arxiv.org/abs/2103.14438.

[Stu+23] Fabian Sturm, Elke Hergenroether, Julian Reinhardt, Petar Smilevski Vojnovikj, and Melanie Siegel. *Challenges of the Creation of a Dataset for Vision Based Human Hand Action Recognition in Industrial Assembly*. 2023. DOI: 10.48550/ARXIV.2303.03716. URL: https://arxiv.org/abs/2303.03716.

[GBV20] Margherita Grandini, Enrico Bagli, and Giorgio Visani. *Metrics for Multi-Class Classification: an Overview*. 2020. DOI: 10.48550/ARXIV.2008.05756. URL: https://arxiv.org/abs/2008.05756.

[Zha+20a] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. "MediaPipe Hands: On-device Real-time Hand Tracking". In: *CoRR* abs/2006.10214 (2020). arXiv: 2006.10214. URL: https://arxiv.org/abs/2006.10214.

[Lin+16] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. "Feature Pyramid Networks for Object Detection". In: *CoRR* abs/1612.03144 (2016). arXiv: 1612.03144. URL: http://arxiv.org/abs/1612.03144.

[Lin+17] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. "Focal Loss for Dense Object Detection". In: *CoRR* abs/1708.02002 (2017). arXiv: 1708.02002. URL: http://arxiv.org/abs/1708.02002.

[Xu+15] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. "Empirical Evaluation of Rectified Activations in Convolutional Network". In: *CoRR* abs/1505.00853 (2015). arXiv: 1505.00853. URL: http://arxiv.org/abs/1505.00853.

[Far+21] Abolfazl Farahani, Behrouz Pourshojae, Khaled Rasheed, and Hamid R. Arabnia. "A Concise Review of Transfer Learning". In: *CoRR* abs/2104.02144 (2021). arXiv: 2104.02144. URL: https://arxiv.org/abs/2104.02144.

[GB04] Yves Grandvalet and Yoshua Bengio. "Semi-supervised Learning by Entropy Minimization". In: *Advances in Neural Information Processing Systems*. Ed. by L. Saul, Y. Weiss, and L. Bottou. Vol. 17. MIT Press, 2004. URL: https://proceedings.neurips.cc/paper/2004/file/96f2b50b5d3613adf9c27049b2a888c7-Paper.pdf.

[Zho+18] Hong-Yu Zhou, Avital Oliver, Jianxin Wu, and Yefeng Zheng. "When Semi-Supervised Learning Meets Transfer Learning: Training Strategies, Models and Datasets". In: *CoRR* abs/1812.05313 (2018). arXiv: 1812.05313. URL: http://arxiv.org/abs/1812.05313.

[Yan+21] Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. "A Survey on Deep Semi-supervised Learning". In: *CoRR* abs/2103.00550 (2021). arXiv: 2103.00550. URL: https://arxiv.org/abs/2103.00550.

[Wen+21] Zejia Weng, Xitong Yang, Ang Li, Zuxuan Wu, and Yu-Gang Jiang. "Semi-Supervised Vision Transformers". In: *CoRR* abs/2111.11067 (2021). arXiv: 2111.11067. URL: https://arxiv.org/abs/2111.11067.

[Xu+21] Yinghao Xu, Fangyun Wei, Xiao Sun, Ceyuan Yang, Yujun Shen, Bo Dai, Bolei Zhou, and Stephen Lin. "Cross-Model Pseudo-Labeling for Semi-Supervised Action Recognition". In: *CoRR* abs/2112.09690 (2021). arXiv: `2112.09690`. URL: `https://arxiv.org/abs/2112.09690`.

[ITP14] Alexandros Iosifidis, Anastasios Tefas, and Ioannis Pitas. "Semi-supervised Classification of Human Actions Based on Neural Networks". In: *2014 22nd International Conference on Pattern Recognition*. 2014, pp. 1336–1341. DOI: `10.1109/ICPR.2014.239`.

[Xia+21] Junfei Xiao, Longlong Jing, Lin Zhang, Ju He, Qi She, Zongwei Zhou, Alan L. Yuille, and Yingwei Li. "Learning from Temporal Gradient for Semi-supervised Action Recognition". In: *CoRR* abs/2111.13241 (2021). arXiv: `2111.13241`. URL: `https://arxiv.org/abs/2111.13241`.

[Ami+22] Massih-Reza Amini, Vasilii Feofanov, Loic Pauletto, Emilie Devijver, and Yury Maximov. *Self-Training: A Survey*. 2022. DOI: `10.48550/ARXIV.2202.12040`. URL: `https://arxiv.org/abs/2202.12040`.

[Lee13] Dong-Hyun Lee. "Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks". In: *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)* (July 2013).

[Pha+20] Hieu Pham, Qizhe Xie, Zihang Dai, and Quoc V. Le. "Meta Pseudo Labels". In: *CoRR* abs/2003.10580 (2020). arXiv: `2003.10580`. URL: `https://arxiv.org/abs/2003.10580`.

[Ber+19] David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. "MixMatch: A Holistic Approach to Semi-Supervised Learning". In: *CoRR* abs/1905.02249 (2019). arXiv: `1905.02249`. URL: `http://arxiv.org/abs/1905.02249`.

[Soh+20] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. "FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence". In: *CoRR* abs/2001.07685 (2020). arXiv: `2001.07685`. URL: `https://arxiv.org/abs/2001.07685`.

[Cas+20] Paola Cascante-Bonilla, Fuwen Tan, Yanjun Qi, and Vicente Ordonez. "Curriculum Labeling: Self-paced Pseudo-Labeling for Semi-Supervised Learning". In: *CoRR* abs/2001.06001 (2020). arXiv: `2001.06001`. URL: `https://arxiv.org/abs/2001.06001`.

[Lu+20] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, João Gama, and Guangquan Zhang. "Learning under Concept Drift: A Review". In: *CoRR* abs/2004.05785 (2020). arXiv: `2004.05785`. URL: `https://arxiv.org/abs/2004.05785`.

[Ben+09] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. "Curriculum learning". In: *International Conference on Machine Learning*. 2009.

[Ber+22] Dan Berrebbi, Ronan Collobert, Samy Bengio, Navdeep Jaitly, and Tatiana Likhomanenko. *Continuous Pseudo-Labeling from the Start*. 2022. DOI: `10.48550/ARXIV.2210.08711`. URL: `https://arxiv.org/abs/2210.08711`.

[Ara+19] Eric Arazo, Diego Ortego, Paul Albert, Noel E. O'Connor, and Kevin McGuinness. "Pseudo-Labeling and Confirmation Bias in Deep Semi-Supervised Learning". In: *CoRR* abs/1908.02983 (2019). arXiv: `1908.02983`. URL: `http://arxiv.org/abs/1908.02983`.

[Zha+20b] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. *Revisiting Few-sample BERT Fine-tuning*. 2020. DOI: `10.48550/ARXIV.2006.05987`. URL: `https://arxiv.org/abs/2006.05987`.

[Dev+18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018). arXiv: `1810.04805`. URL: `http://arxiv.org/abs/1810.04805`.

[HR18] Jeremy Howard and Sebastian Ruder. "Fine-tuned Language Models for Text Classification". In: *CoRR* abs/1801.06146 (2018). arXiv: `1801.06146`. URL: `http://arxiv.org/abs/1801.06146`.

[Zha+19] Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank J Reddi, Sanjiv Kumar, and Suvrit Sra. *Why are Adaptive Methods Good for Attention Models?* 2019. DOI: 10.48550/ARXIV.1912.03194. URL: https://arxiv.org/abs/1912.03194.

[LTL19] Jaejun Lee, Raphael Tang, and Jimmy Lin. “What Would Elsa Do? Freezing Layers During Transformer Fine-Tuning”. In: *CoRR* abs/1911.03090 (2019). arXiv: 1911.03090. URL: http://arxiv.org/abs/1911.03090.

[Wan+18] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime G. Carbonell. “Characterizing and Avoiding Negative Transfer”. In: *CoRR* abs/1811.09751 (2018). arXiv: 1811.09751. URL: http://arxiv.org/abs/1811.09751.

[PY10] S.J. Pan and Q. Yang. “A Survey on Transfer Learning”. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10 (2010), pp. 1345–1359.

[Ros+05] Michael Rosenstein, Zvika Marx, Leslie Kaelbling, and Thomas Dietterich. “To Transfer or Not To Transfer”. In: *NIPS 2005*. Jan. 2005.