Research on Complex Activity Recognition in Industrial Domains

Submitted to
Graduate School of Information Science and Technology
Osaka University

January 2023

Naoya YOSHIMURA

List of Publication

1. Journal Paper

- Naoya Yoshimura, Takuya Maekawa, Daichi Amagata, Takahiro Hara, "Upsampling Method for Wearable Inertial Sensors with Neural Networks", Journal of Information Processing (Vol.61, No.8), August 2020. (in Japanese)
- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, "Visualization of Deep Feature Representation Toward Transfer Learning of Activity Recognition Models", Journal of Information Processing (Vol.62, No.5), May 2021. (in Japanese)
- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, Atsushi Wada, Yasuo Namioka, "Acceleration-based Activity Recognition of Repetitive Works with Lightweight Ordered-work Segmentation Network," In Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (Vol.6 No.2), July 2022.
- Kei Tanigaki, Teoh Chuin Tze, Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, "Predicting Performance Improvement of Human Activity Recognition Model by Additional Data Collection," In Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies (Vol.6 No.3), September 2022.

2. International Conference Paper

- Naoya Yoshimura, Takuya Maekawa, Daichi Amagata, Takahiro Hara, "Preliminary Investigation of Fine-grained Gesture Recognition with Signal Super-resolution," In Proceedings of IEEE International Conference on Pervasive Computing and Communications Workshops, pp.484–487, January 2018. (WiP Paper)
- Naoya Yoshimura, Hironori Yoshida, Fabrice Matulic, Takeo Igarashi, "Extending Discrete Verbal Commands with Continuous Speech for Flexible Robot Control," In Proceedings of Extended Abstracts of the CHI Conference on Human Factors in Computing Systems, pp.1–6, May 2019. (LBW Paper)

- Naoya Yoshimura, Takuya Maekawa, Daichi Amagata, Takahiro Hara, “Upsampling Inertial Sensor Data from Wearable Smart Devices Using Neural Networks,” In Proceedings of IEEE International Conference on Distributed Computing Systems, pp.1983-1993, July 2019.

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “Preliminary Investigation of Visualizing Human Activity Recognition Neural Network,” In Proceedings of International Conference on Mobile Computing and Ubiquitous Networking, pp.1-2, November 2019. (Poster Presentation)

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “Toward Understanding Acceleration-based Activity Recognition Neural Networks with Activation Maximization,” In Proceedings of International Joint Conference on Neural Networks, Virtual, July. 2021.

- Naoya Yoshimura, Yushi Sato, Yuta Kageyama, Jun Murao, Satoshi Yagi, and Parinya Punpongsanon, “Hugmon: Exploration of Affective Movements for Hug Interaction using Tensegrity Robot,” In Proceedings of ACM/IEEE International Conference on Human-Robot Interaction, Virtual, March. 2022. (LBW Paper)

- Jaime Morales, Naoya Yoshimura, Qingxin Xia, Atsushi Wada, Yasuo Namioka, Takuya Maekawa, “Acceleration-based Human Activity Recognition of Packaging Tasks Using Motif-guided Attention Networks,” In Proceedings of IEEE International Conference on Pervasive Computing and Communications, March 2022.

3. Domestic Conference Paper

- Naoya Yoshimura, Takuya Maekawa, Daichi Amagata, Takahiro Hara, “ジェスチャ・行動認識のための加速度信号アップサンプリング手法に関する検討,” IPSJ SIG-UBI Technical Reports, May 2018. (in Japanese)

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “行動認識モデルの転移学習に向けたニューラルネットワークによる特徴抽出の可視化と分析,” Multimedia, Distributed, Cooperative, and Mobile Symposium, July 2019. (in Japanese)

- Jaime Morales, Naoya Yoshimura, Qingxin Xia, Takuya Maekawa, Atsushi Wada, Yasuo Namioka, “Preliminary investigation on activity recognition for packaging tasks using motif-guided attention networks,” IPSJ SIG-UBI Technical Reports, pp.1-8, March 2020.

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “行動認識ニューラルネットの理解に向けたActivation Maximizationの活用に関する検討,” IPSJ SIG-UBI Technical Reports, December 2020. (in Japanese)

- Kei Tanigaki, Teoh, Chuin, Tze, Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “行動認識モデル構築における追加学習データ収集支援のための認識精度向上予測に関する検討,” IPSJ SIG-UBI Technical Reports, November 2021. (in Japanese)

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, Atsushi Wada, Yasuo Namioka, “工場・物流センタにおける作業順序を考慮した少量学習データでの作業行動認識手法の検討,” IPSJ SIG-UBI Technical Reports, June 2022. (in Japanese)

- Naoya Yoshimura, Jaime Morales, Takuya Maekawa, Takahiro Hara, “OpenPackデータセット: 物流センタにおける 大規模マルチモーダル行動認識データセットの構築,” IPSJ SIG-DPS Technical Reports, December 2022. (in Japanese)

4. Award

- Naoya Yoshimura, Takuya Maekawa, Daichi Amagata, Takahiro Hara, “Best WiP Paper Award,” IEEE International Conference on Pervasive Computing and Communications, March 2018.

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “優秀論文賞,” Multimedia, Distributed, Cooperative, and Mobile Symposium, July 2019.

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “Best Poster Award,” International Conference on Mobile Computing and Ubiquitous Networking , October 2019.

- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “優秀論文賞,” IPSJ SIG-UBI, Decemver 2020.
- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, “特選論文,” IPSJ, May 2021.
- Naoya Yoshimura, “山下記念研究賞,” IPSJ, March 2022.
- Naoya Yoshimura, Takuya Maekawa, Takahiro Hara, Wada Atsushi, Yasuo Namioka, “優秀論文賞,” IPSJ SIG-DPS, June 2022.

5. Grant

- “ACT-X,” Japan Science and Technology Agency, November 2020.
- “DC2,” Japan Society for the Promotion of Science, April 2021.

Abstract

Many manufacturers have introduced robots, sensors, and IoT devices into their industrial sites, and used the collected data from them to improve the productivity. Specifically, sensors that monitor manual works by workers are expected to be used for incorporating the real-world work activity information into the cyberspace, enabling to simulate manufacturing processes with machine and human generated data. Although sensor data are now easily obtained by attaching a smartwatch or other sensors to a human worker, the data represent just a sequence of numerical values, not high-level information that describes the work activities. Therefore, the development of work activity recognition technologies that transform sensor data into high-level work activity information is required.

In this thesis, we aimed to address three challenges for the development of work activity recognition models and their practical applications. The first challenge is the amount of training data. The cost of collecting labeled training data to build a work activity recognition model for each worker in a factory is huge, and it is necessary to build the model with a limited amount of training data to reduce this cost. The second challenge is the usage of data from IoT devices. There are many IoT devices deployed in the site, which can be an important source of information for work activity recognition. However, there is no dataset of for work activity recognition that includes both sensor data and data from IoT devices. Due to the unique characteristics of IoT device data, it is necessary to explore the ways to combine sensor data with IoT device data that take into account the characteristics of IoT device data. The third challenge is the interpretability of neural network-based work activity recognition models. Neural networks are considered to be black boxes, but it is important to improve the interpretability of the models in order to introduce the models into the actual industrial sites.

Different workers on a production line are assigned to different tasks. In addition, workers may change the standard work processes depending on their skill level and other factors. Therefore, it is difficult to build a work recognition model for a target worker by using training data collected from other workers. We proposed a lightweight ordered-work segmentation network (LOS-Net), which is a lightweight model that extracts minimal necessary features and can be trained on limited training data from the target worker. We evaluated LOS-Net using data collected from 11 actual factory workers and found that LOS-Net improved recognition performance by about 10% on average (F1-measure macro average) compared to baseline methods.

Many IoT devices have been installed in industrial sites. The usage logs of the IoT devices strongly relates to the user's activities and can be a very important source of information for work activity recognition. However, because the data is generated only at the time when an IoT device is used, it is much sparse in time compared to general sensor data such as acceleration data. Therefore, a method to effectively fuse sensor data with IoT data should be explored. To accelerate activity recognition studies based on the sensor and IoT data fusion, we developed a large-scale multimodal dataset of packing work activity called the "OpenPack dataset." We collected over 53 hours of work activity data from 16 subjects and obtained 20,129 work operation labels and 52,529 action labels. This dataset provides a total of nine data modalities, including sensor data from inertial measurement units (IMUs), depth images, and keypoints, as well as usage logs from IoT devices. In addition, we proposed a method that effectively fuses acceleration data with IoT device data. We evaluated the model on the OpenPack dataset and improved the F1-measure (macro average) by 0.03 pt compared to existing methods.

In addition to the development of work activity recognition methods, we also addressed the issue of the interpretability of activity recognition neural networks. Many recent activity recognition models, including models developed in this study, rely on neural networks. There are few line managers who are familiar with ICT, and it is not easy for them to trust the results of neural networks, which are considered as black boxes. In this study, we applied a method called activation maximization to the sensor-based activity recognition model in order to make the features extracted by the neural network interpretable. Activation maximization is a method of visualizing feature representations extracted by a unit in a neural network by generating an input signal to which the unit of interest strongly responds. In generating such an input signal, the design of the regularization term is important. We proposed two regularization methods that use the unit outputs to determine if anomalous waveforms are generated and penalize them accordingly. Qualitative and quantitative evaluations of the generated waveforms were conducted to confirm their effectiveness.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Many manufacturers are now introducing robots, sensors, and IoT devices into their factories aiming to transform them into smart factories. The data obtained from these equipments are expected to be integrated and analyzed on the Cyber-Physical System (CPS) and digital twin, and used for factory optimization such as factory monitoring, finding bottlenecks of a work process, and improving the productivity as shown in the upper portion of Fig. 1.1.

In the CPS and digital twin, it is important to reconstruct a copy of a physical factory in the cyberspace as accurately as possible, and the accuracy of the reconstruction in cyberspace directly affects the accuracy of the simulation and analysis. From software-controlled equipment such as machines and robots, control commands and working status can be obtained directly via electronic means. This makes these machines compatible with the CPS and digital twin.

As an application of the CPS and digital twin, RPA (robotic process automation) uses AI and other technologies to replace human workers. RPA is attracting attention as a solution to the shortage of workers in manufacturing and other industries as well as to maintain the product quality while reducing costs. However, it is not easy to replace all the workers with robots and machines in the manufacturing process. In recent years, the high-mix low-volume production is the mainstream to flexibly respond to the diverse and changing demands of customers. It is not cost-effective to introduce ex-

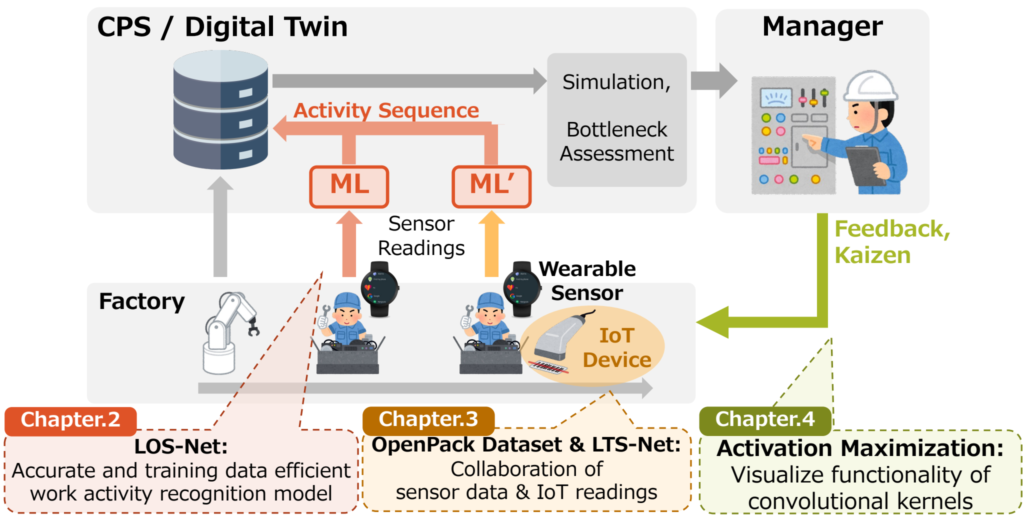


Figure 1.1: Overview of this study

pensive robotic systems into the production system because tasks to conduct frequently change, which require additional development and installation costs. Therefore, it is expected that these tasks such as assembly work in production lines and packaging work at distribution centers will rely on human workers.

While human workers can perform complex and flexible tasks, it is not easy to obtain information about their working status unlike manufacturing machines. This means that it is still difficult to accurately incorporate human works into the simulations in the CPS and digital twin. Although we can collect data from workers by attaching wearable devices, such as smartwatches, to the workers, the collected sensor data, which are just series of numerical data, do not describe high-level real-world contexts such as activities of workers. Therefore, it is required to develop an activity recognition technology that converts sensor data into high-level information, i.e., work activities [1, 95, 94, 52, 2, 69, 21].

Activity recognition technologies use machine learning and other techniques to analyze data obtained from a sensing device, such as a smartwatch worn by a user, to estimate what kind of activity the user is performing for each time step. Many existing studies focus on the activities of daily living (ADL) and exercise such as "waking," "jumping," "ascending stairs," and "folding laundry" [71, 9, 7]. In this study, we focus on work activities instead of such daily life activities. Specifically, we assume work activities such as "installing screws," "attaching labels," and "scanning barcodes" in assembly work in a production line and packing work at a logistics center. By applying activity recognition technologies to work activities, it is possible to know which worker is performing which work activity at each time step, and how much time the worker spends for each activity. This information is important for optimizing the manufacturing process via the simulation in the CPS/digital twin. For example, when the end of each work activity is accurately captured, it is possible to know when and how much idle time is available, and to adjust the amount of tasks to be allocated to each worker. Ultimately, we can open the way to the optimization of the entire work process in an industrial site by taking into account not only the machines but also the human factors that play a crucial role in the site.

Although camera-based activity recognition techniques have been actively studied [97, 80, 96], the vision-based methods require multiple computationally expensive processes other than activity recognition, such as the detection and identification of workers. They also require high data transmission costs and a large data storage when captured images are processed on a server computer with GPUs. Furthermore, the problem of occlusion cannot be ignored because of the variety of objects placed in an industrial site. In contrast, because methods relying on wearable devices do not require the detection and identification of workers, the developers can concentrate on the action recognition task. In addition, the model is relatively lightweight compared to the image-based methods, making it easy to address the scalability issue when applying the activity recognition technology to all workers in the site. In terms of installation costs, the cost of a camera-based behavior recognition system is expected to be very expensive. This is because multiple cameras are required per worker to avoid occlusion. In addition, high-specification cameras are required in factories to ensure reliable performance in environments with extreme temperatures and humidity. The cost per camera can rival or exceed the cost of purchasing a smartwatch. Moreover, processing of video stream data requires high-performance GPU cards, which is an expensive cost. For these reasons, we focused on the use of wearable devices in this study.

The characteristics of activity recognition technologies for factories and logistics centers differ significantly from those for activities of daily living. First, **different workers are assigned to different tasks in many cases**, and even the same task is performed in different ways depending on the workers' skill levels. Therefore, it is necessary to develop an activity recognition model for each worker. In other words, it is not easy to use training data collected from different workers as training data for another worker because of the above feature of the work activities. In addition to this, we need to repeatedly update a large number of models due to the changes in the production process, for example. Second, **there are various business systems installed in factories and logistics centers** that manage data of IoT device usage and operations. These data contain information strongly related to the worker's activities, which have a potential to improve the work activity recognition performance. Third, when making a decision regarding work processes based on the results of activity recognition, **it is necessary to obtain the confidence of the production line manager regarding the results of activity recognition**. Specifically, a process in a deep model is considered to be a black box, making it difficult to understand the process by humans. Currently, because there are a few managers in industrial sites who are knowledgeable about ICT, it is not easy for the managers to make a decision based on the recognition results from the black box, and this can be a barrier to the introduction of activity recognition technologies.

As described above, the characteristics of work activities differ significantly from those of activities of daily living, and there are many challenges that need to be addressed. However, by solving these problems, it will be possible to utilize information about human workers which are considered to be highly uncertain. As a result, overall optimization of the industrial sites with consideration of human factors will be achieved.

1.2 Challenges

As mentioned above, activity recognition technologies in the industrial domains have a wide range of possible applications. However, the work activities differ significantly from the activities of daily life on which many existing studies have focused. In this section, we explain the challenges arising from these differences.

1.2.1 Work Processes Differ with Workers

In an operation flow in a production line, a production process is divided into multiple steps, and workers at different steps perform different tasks. In addition, skilled workers usually modify standard work processes and have their own ways of doing things. In order to monitor the performance of workers throughout a factory, it is necessary to build a large number of different activity recognition models for different workers because it is not easy to use the training data collected from one worker as training data for other workers because of the above features of the work activities. Therefore, it is inevitable to collect some training data from each worker.

However, the cost of collecting labeled training data is large [95], and this cost is not acceptable when the industrial site employs many workers. Therefore, the amount of data collected from each worker should be small, and models should be trained on the limited amount of training data.

1.2.2 Difficulties in Sensor–IoT Data Coordination

Various IoT devices are installed in smart factories and logistics centers. A usage log of an IoT device strongly relates to the user's activity at that time. For example, the usage log generated when a user scans a barcode with a scanner indicates that the user performed the "scan" action at that time with high probability, and is a reliable source of information for activity recognition.

In contrast, the usage log of an IoT device is generated only at the time of use. For example, when considering the action of scanning a label, it is reasonable to consider the process from holding the scanner to putting the scanner on the table after reading the barcode as a single activity, but the usage history of the scanner is recorded only at the moment when the scanner recognizes the barcode. In other words, compared to sensor data such as those from an accelerometer, which constantly generates data at regular intervals, IoT device data is highly reliable information for recognizing specific actions but is very sparse in time.

Therefore, the use of IoT device data, which is very different from sensor data, in conjunction with general sensor data should be well investigated [54]. However, although many datasets of daily activities are publicly available [71, 68], work activity datasets that include both sensor data and IoT data are not publicly available, making it difficult to explore the efficient sensor–IoT data coordination.

1.2.3 Deep Learning Model is a Black Box

At present, few factory managers are familiar with ICT and related technologies. Besides, many state-of-the-art (SOTA) methods in recent activity recognition research [109, 64, 82, 32] are based on deep neural networks. Although neural network-based methods tend to have high recognition accuracy, their recognition processes, which are regarded as a black box, are more difficult to understand than those of the classic methods that require manual feature extraction [65, 44].

In the case of exercise and sleep stage recognition, for example, users use the systems to browse their activity histories, i.e., activity recognition results. The recognition results are used only as a reference and are rarely used to make important decisions. Therefore even when the recognition algorithm is a black box, it can be used without any problems as long as the recognition is accurate.

In contrast, the factory manager makes decisions based on activity recognition results that may cause economic losses. Therefore, it is difficult to make important decisions based on unreliable algorithms, and there is a demand for understanding the black box process.

1.3 Approach

This study addressed the three issues in work activity recognition described in the previous section as follows.

1.3.1 Recognize Work Operations of Workers with a Limited Amount of Training Data

In Chapter 2, we developed a work activity recognition model that enables recognition with a limited amount of training data collected from a target worker. We proposed the Light-weight Ordered-work Segmentation Network (LOS-Net), which is a compact model designed to extract minimum features, taking into account the characteristics of work activities. To realize this model, we proposed three modules included in LOS-Net.

First, the work process context pooling (WPCP) module, which uses a dilated convolution [28, 88, 87, 14] to efficiently capture features from a wide range was proposed.

By combining multiple dilated convolutions, it can efficiently extract the short-term and long-term context that is important for work process recognition. Second, the boundary detector was designed for accurately detecting the start and end times of a work activity. Since work activity recognition is a segmentation task, when the start and end times are accurately detected, we can know activity labels between the two times. Because the boundary of the activity itself depends on the previous and following work activities, recognition becomes difficult when the order of the work activities is changed due to errors or other reasons. Therefore, the system should be robust against changes in the order of work activities by estimating the start and end times of each work activity independently. The third module is the refinement module. Because the standard work procedures are specified, the order of the work activities is basically static. The refinement module uses prior knowledge about the standard work activity order to correct sporadic errors included in estimates by the recognition model.

1.3.2 New Large-scale Multimodal Dataset in Industrial Domain

In Chapter 3, to solve the problem of the lack of work activity datasets that include both sensor data and IoT device data, we constructed the OpenPack dataset, a large-scale multimodal dataset in the industrial domain that focuses on packaging work. OpenPack contains over 53 hours of data from a total of 16 subjects, including 12 subjects who had work experience in packaging. In addition to IoT device data, nine types of data were collected, including IMU data, depth images, keypoints, and stress-related data such as user blood volume pulse (BVP) and electrodermal activity (EDA). We annotated work activities (operations) and their atomic actions, resulting in 20,129 operation labels and 52,529 action labels.

We also studied how to effectively fuse data with different characteristics, such as sensor data from IMUs and data from IoT devices, and proposed the "Ladder-Shaped Two-Stream Network (LTS-Net)." We can precisely estimate an activity label only at the time when a log of an IoT device is recorded, meaning that IoT device logs are high-confidence information while sparse in time. However, existing sensor fusion methods [54, 30, 31] are not designed for the sparse data and cannot guarantee that this highly reliable information will remain in the output because all the data are fed into the neural network and recomputed in the network. Therefore, LTS-Net first generates pseudo-labels by using data from IoT devices, which are regarded as reliable estimates, and updates the pseudo-labels with the help of sensor data while preserving high confidence pseudo-labels. To achieve this, we prepare a module for feature extraction from sensor data and a module for updating the pseudo-labels. These two modules exchange information each other to form a ladder-shaped structure. The module that updates the pseudo-labels has no trainable parameters, but updates the estimates according to the confidence level of the pseudo-labels, enabling to preserve highly reliable information from the IoT devices.

1.3.3 Visualization of the Concept Learned by a Unit of the Neural Network

In Chapter 4, we applied activation maximization [19] to an activity recognition model to unveil the process of the recognition model regarded as a black box. Activation maximization is a method to visualize the concepts extracted by each unit in the neural network. Activation maximization generates an input signal in which the unit of interest emits a large value (activation), enabling to visualize a waveform the unit of interest responds to. However, in the waveform generation method, the design of an appropriate regularization method greatly affects the quality of the generated waveforms [40, 105, 55]. Without an appropriate regularization method, waveforms with very large amplitudes that are not found in real situations are generated. In this study, we propose a regularization method that uses the unit outputs to determine if anomalous waveforms are generated and penalize them accordingly. We also created an "activation atlas" to obtain a bird's-eye view of the extracted features, and conducted a qualitative evaluation.

1.4 Research Contribution

In this thesis, we make the following contributions.

In Chapter 2:

- We proposed a new activity recognition network (called LOS-Net) for ordered repetitive works in industrial domains. The proposed network is designed to be trained on a limited amount of training data because preparing large training data in industrial settings is expensive. LOS-Net has the following features compared to conventional activity recognition models: (i) The network includes a decoder with few parameters designed to capture only the necessary information for precise segmentation/recognition: long-term context regarding ordered works and information regarding the boundaries between consecutive activities. (ii) The network can refine the outputs of the decoder by referring to prior information regarding the predefined order of activities. (iii) Owing to the above features, LOS-Net works well with a limited amount of training data, which was investigated by experiments.

- We demonstrated the effectiveness of the proposed method using sensor data collected from actual factories and an actual logistic center.

In Chapter 3:

- We constructed OpenPack, which is the largest dataset on industrial work activity recognition. We expect this data to contribute to research on industrial AI systems by providing a challenging task, that is, to complex activity recognition via sensor and sparse IoT data.

- We proposed LTS-Net to effectively utilize sparse readings from IoT devices via a ladder structure.

- The results of the experimental evaluation showed that LTS-Net significantly improved the performance, indicating the usefulness of the readings from IoT devices and the effectiveness of the ladder structure in our method.

In Chapter 4:

- We proposed new activation maximization (AM) techniques for human activity recognition networks: (i) a regularization term that penalizes extreme activations generated in the AM process, and (ii) activation clipping that suppresses the generation of extreme activation. By leveraging the proposed techniques, we can generate natural signals with reasonable activation values.

- We evaluated the proposed method qualitatively and quantitatively using publicly available datasets, and it was found that the proposed method could generate signals that are similar to those that actually appeared in a training dataset.

1.5 Structure of the Thesis

This thesis is structured as follows.

- *Chapter 1 - Introduction*
  We first provided an introduction to this thesis. We discussed the background of smart industrial environments and work activity recognition using wearable devices such as smartwatches.

- *Chapter 2 - Acceleration-based Activity Recognition of Repetitive Works with Lightweight Ordered-work Segmentation Network*
  We introduce a possible solution for recognizing work activities with limited training data. We proposed a model that extracts only necessary information from sensor data and utilizes prior knowledge. The study in this chapter is based on our work published in [103].

- *Chapter 3 - OpenPack: A Large-scale Dataset for Recognizing Packaging Works in IoT-enabled Logistic Environments*
  We introduce a new large-scale dataset in industrial domain. This dataset contains data from IoT devices in addition to sensor and vision modalities. We also proposed a model that fuses sensor data and IoT data effectively. The study in this chapter is based on our work published in [104].

- *Chapter 4 - Toward Understanding Acceleration-based Activity Recognition Neural Networks with Activation Maximization*
  We introduce a visualization technique that reveals the concepts extracted by human activity recognition models. A neural network is regarded as a black box, and this method is one of the possible solutions for this problem. The study in this chapter is based on our work published in [102].

- *Chapter 5 - Conclusion and Future Work*
  We conclude this thesis with a summary of contributions and with a discussion of possible directions for future works.

Chapter 2

Acceleration-based Activity Recognition of Repetitive Works with Lightweight Ordered-work Segmentation Network

In this chapter, we describe our work on recognizing work activities using wearable sensors. We explored the methods that can utilized prior knowledge such as activity order to develop a model with a limited amount of training data.

2.1 Introduction

2.1.1 Background

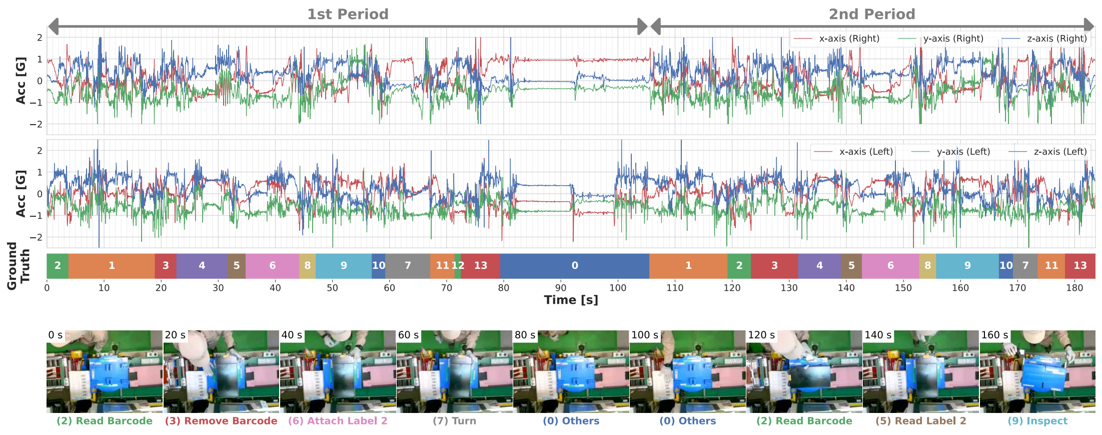
Activity recognition using smart sensing devices such as smartphones and smartwatches has been actively studied by the ubiquitous computing research community. Human activity recognition, besides for daily life settings [71, 9, 7], has been studied for industrial domains to improve the efficiency of manual works [1, 95, 3, 94, 2, 69, 21]. Work processes such as those in the line production systems of factories and packaging tasks at logistics centers mainly depend on human workers. To deal with the rapidly changing demands of customers and suppliers, works by human workers are expected to continue to play an important role in the future [35, 66, 99]. Therefore, quantifying such manual works is crucial for streamlining a work process, finding its bottlenecks, assessment of a worker's performance, and outlier detection.

In many manual works, such as those in line production systems and packaging tasks in the logistic domain, a human worker repetitively performs a set of predefined processes, with each process consisting of a sequence of activities in a predefined order, e.g., attach labels, scan labels, and check size and weight. Fig. 2.1 shows example acceleration data collected from the smartwatches worn by workers at an actual factory. In this example, a typical series of works is iterated several times, with each iteration of the process (i.e., work period) comprising a sequence of activities. Although the order of activities is specified (by an instruction document in the case of factory assembly works), the order can be changed by some factors, such as mistakes. The objective of an activity recognition task focusing on such data is to estimate the class label for each time step (data point).

2.1.2 Challenges

The major challenge of this task is the preparation of a large amount of training data in actual industrial domains, because of the high costs of installing cameras at a work place for annotation and manual labeling of the collected data. Moreover, in many cases, the work processes performed in factories differ with the worker, making it difficult to reuse labeled training data obtained from different workers. For example, in a factory line production system, workers positioned at different stages along the line usually conduct different predefined activities. This means that it is impossible to use training data from other workers on the same line to train an activity recognition model of a target worker. Whereas ultra-large factories have multiple lines for the same product, high-mix low volume production is now the mainstream production type to deal with rapid changing demands of consumers, resulting in different activities among different lines in many cases. Therefore, activity recognition using a limited amount of training data from a target worker is crucial in industrial domains.

The architecture of the state-of-the-art human activity recognition methods using body-worn acceleration sensors includes an encoder–decoder with skip connections, i.e., a U-Net-based architecture [72]. The encoder has multiple convolutional and pool-



(a) Example of sensor data

(b) Activities that Shown in the Above Panel

ID	Activity Name	ID	Activity Name	ID	Activity Name	ID	Activity Name	ID	Activity Name
0	Others	3	Remove Bar Code	6	Attach Label 2	9	Inspect	12	Button
1	Set	4	Attach Label 1	7	Turn	10	Remove Jig	13	Move
2	Read Bar Code	5	Read Label 2	8	Attach Jig	11	Weighing		

Figure 2.1: Example of sensor data of factory works. Red, green, and blue lines show x-, y-, and z-axis data, respectively. Rectangles below show ground truth labels (identifiers) of activities. Top-down view video images are also shown. Table provides identifier and name of each activity.

ing layers and gradually down-samples the input time series (sensor data) to acquire high-dimensional representations with a low temporal resolution. Subsequently, the decoder containing multiple deconvolution layers up-samples the low-resolution representations to output the class label for each time step. U-Net is used for image semantic segmentation and has the advantage of capturing long-term trends in sensor-based human activity recognition. However, because the encoder and the decoder comprise numerous trainable nodes, the architecture requires large training data. Fig. 2.2 shows a distribution of the recognition accuracy of U-Net for factory work when we performed the validation, where a set of 10-period data were used as training data[1]. As the amount

[1]Obtained periods were divided into sets of periods, with each set being composed of ten periods. Each set was used as a test set and the other set was used as a training set. The training was conducted

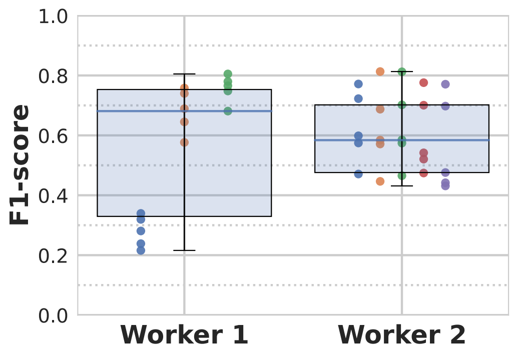


Figure 2.2: Distribution of recognition accuracy of U-Net with limited training data (10 periods). Dots of the same color show F1-measures of the same test and training sets with different random seeds.

of training data is limited, the accuracy of U-Net, which contains large trainable parameters, is unstable. Herein, a trainable parameter indicates a parameter of a neural network such as the weight of a node in a densely connected layer and a weight of a kernel in a convolution layer. Even when we used the same test and training data, the accuracy largely varied depending on random seeds, resulting in unreliable recognition systems.

Besides, activities that are almost performed in a predefined order can be recognized by leveraging the information regarding their order as prior information. For example, several previous studies on activity recognition using classic machine learning specify the transition probabilities between activities as prior information [18, 62, 47]. In the case of human activity recognition using deep models, we can leverage recurrent neural networks, such as long short-term memory (LSTM), to learn the long-term trends in ordered activities. However, this approach also requires a large amount of training data.

2.1.3 Approach

In this study, we propose a human activity recognition model that can be trained on a limited amount of training data from a target worker who is engaged in repetitive works involving a predefined order of activities. Our idea to achieve work activity recognition

five times by changing random seeds. Refer to the evaluation section for the architecture of U-Net.

with limited training data is to capture only the necessary information for precise activity recognition, with a lightweight model. As shown in Fig. 2.1, an acceleration segment for each activity is complex and composed of various waveforms (corresponding to small hand actions). To precisely recognize such activities, it would be ideal to accurately detect all these small actions. However, this approach requires a large model and a large amount of training data. As this task, i.e., recognizing sequential activities, can be regarded as a semantic segmentation task, we note that it is not necessary to precisely identify small actions that are performed in the midst of an activity of interest. Instead, it is more important to precisely detect small actions performed at the beginning and end of an activity. In other words, it is only necessary to roughly identify actions performed in the midst of the activity. In addition, for the recognition of ordered tasks, information regarding the previous activity of an activity of interest is an important indication that enables the precise detection of the beginning of current activity of interest. When we can acquire long-term contextual information regarding the previous activity (e.g., information regarding a sequence of small actions in the previous activity), we can robustly identify the beginning of the following activity of interest. For example, when small actions A, B, and C are usually performed in this order at the end of the previous activity, we can use that information to identify the start time of the activity of interest.

Therefore, in this study, for precise segmentation/recognition using limited training data, we propose a neural network architecture (called Lightweight Ordered-work Segmentation Network; LOS-Net) that performs time-series segmentation/recognition using a decoder that captures only necessary information. Specifically, a feature of the decoder of the LOS-Net is performing segmentation/recognition by fusing the following two types of important information with few parameters: (i) To achieve precise segmentation, precise detection of the boundary between consecutive activities is crucial. As shown in Fig. 2.1, we can observe trend changes in sensor data at some activity boundaries (e.g., boundary between the fourth and fifth activities). Therefore, we extract information on boundaries mainly from representations acquired in the shallow layers of the LOS-Net encoder, which are expected to preserve the fine-grained information on boundaries, and subsequently use the extracted boundary information to perform segmentation/recognition. (ii) Long-term context of the work process of interest is crucial for recognizing *ordered* activities. As mentioned above, in the recognition of an activity, the information regarding its previous activity (e.g., information regarding a sequence of small actions in the previous activity) is useful. Therefore, we extract long-term context from low-resolution representations acquired in the last layer of the encoder using a convolution layer with few parameters. In the convolution layer, we apply a dilated convolution [28, 88, 87, 14], which inflates a convolution kernel by inserting holes between the kernel elements, enabling a large receptive field and collection of information from distant time steps using few parameters. Subsequently, we up-sample the low-resolution long-term context extracted by the dilated convolution by simply using linear interpolation, enabling to up-sample the representations with no trainable nodes. (In contrast, U-Net gradually up-samples the low-resolution representations using many deconvolution layers, thus requiring large parameters.)

Following this, we fuse the representations regarding the boundaries and the long-term context to predict the activity class for each time step. As mentioned above, we extract minimal important information (i.e., information regarding boundaries and the long-term context) for precise segmentation/recognition using the decoder with few parameters.

To further enhance segmentation precision, the LOS-Net includes a module that refines the outputs of the decoder by incorporating prior knowledge regarding the order of activities, which can also accelerate precise recognition with limited training data. In the work process in a line management system, the order of activities is generally predefined in an instruction document. In packaging tasks, the order of activities is also predefined, e.g., making a box followed by filling it with items. In this study, we assume that the activity order is provided as a transition matrix, and subsequently correct the outputs of the decoder based on the matrix. For example, we assume a work process composed of activities A, B, C, D, and E performed in the same order. We also assume that the decoder outputs show that activity E is performed immediately after activity A. We detect such outlying transitions between the activities (i.e., A–E) by referring to the transition matrix and subsequently refine the estimates in the segment corresponding to the transition. In addition, a refinement procedure is designed to address the insufficiency of the training data by data augmentation.

2.1.4 Contributions

The research contributions of this study are as follows:

- We propose a new activity recognition network (called the LOS-Net) for ordered repetitive works in industrial domains. The proposed network is designed to be trained on a limited amount of training data because preparing large training data in industrial settings is expensive. The LOS-Net has the following features compared with conventional activity recognition models: (i) The network includes a decoder with few parameters designed to capture only the necessary information for precise segmentation/recognition: long-term context regarding ordered works and information regarding the boundaries between consecutive activities. (ii) The network can refine the outputs of the decoder by referring to prior information regarding the predefined order of activities. (iii) Owing to the above features, the LOS-Net works well with a limited amount of training data, which was investigated by experiments.

- We demonstrate the effectiveness of the proposed method using sensor data collected from actual factories and an actual logistic center.

2.2 Related Work

In recent years, with growing health awareness, wearable devices such as smartwatches have become increasingly prominent and are used to record exercise, sleep, and vitals (such as heartbeat) data. Wearable devices are equipped with various sensors, such as inertial measurement units, heart rate sensors [68], electrodermal activity sensors [25, 50, 92], and microphones [24, 23]. Research on activity recognition using data obtained from such sensors has been actively conducted in the field of ubiquitous computing. Particularly, accelerometers have been gaining significant interest in many activity recognition studies because they can directly capture details of the movements of the body parts to which they are attached. We also use accelerometers in this research.

In this section, we summarize the applications of activity recognition technologies based on accelerometers, activity recognition methods, and industrial applications of activity recognition.

2.2.1 Acceleration-based Human Activity Recognition

In the field of activity recognition using accelerometers, there have been studies to recognize activities of daily living (such as cleaning and cooking), exercises (such as running and workout), and activities of workers in factories and hospitals [68, 8, 9, 5, 94, 29].

Recently, activity recognition methods using neural networks have been actively studied. Some methods use recurrent models, such as LSTM and gated recurrent units (GRU) [26, 107], and others employ one-dimensional convolutional neural networks (CNNs) [26, 54, 60]. In addition, methods that combine CNNs and LSTM have been studied. In particular, DeepConvLSTM proposed by Ordóñez et al. [63] has been used as a baseline in many activity recognition studies. In this study, we also use it as a baseline for comparison. In this study, we apply a dilated convolution to efficiently extract the long-term context in ordered activities using few parameters.

To recognize activities using acceleration data, many studies apply a sliding time window and predict the activity class for each window [26]. In contrast, many human works in industrial domains, which are the focus of this research, are composed of a mixture of short- and long-duration activities performed sequentially without intervals. Recognizing such activities using a time window with a fixed window size is difficult. For example, a small-sized window that fits short-duration activities is ineffective for long-duration activities because of their fragmentation in the sliding time window scheme.

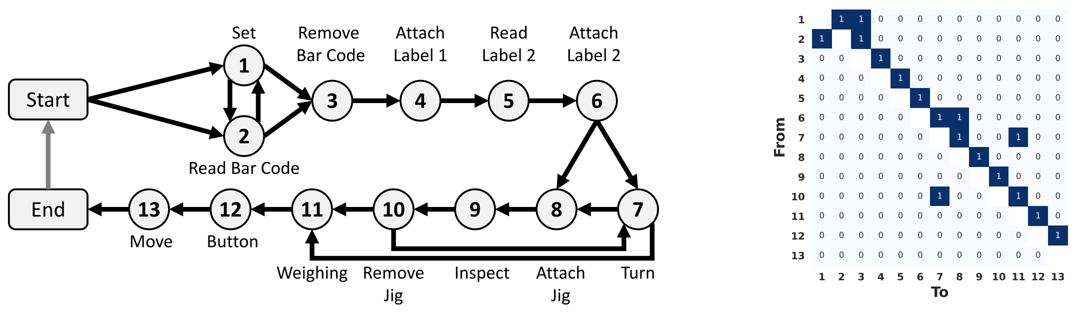
To address the above problem, Yao et al. [98] proposed an approach called segmentation or dense labeling, which predicts the class label of each data point in the input time series. They also proposed a segmentation model motivated by a fully connected network [42]. Following this study, a segmentation model based on U-Net was proposed [109, 108]. Zheng et al. [109] used a U-net-based model for activity recognition. In their study, they pointed out that a segmentation model generates small errors called fragmentation and substitution. They proposed a method to reduce these errors by rule-based post-processing. A U-Net based approach for time-series segmentation is also used in the sleep stage recognition task [64]. In this study, we propose a lightweight segmentation model based on an encoder–decoder model. In addition, we propose a postprocessing method by leveraging the predefined activity order.

2.2.2 Human Activity Recognition in Industrial Domains

Activity recognition technologies using wearable devices have been studied for applications in work environments such as factories, logistics centers, hospitals, and nursing homes. In these domains, various applications have been developed based on activity recognition, such as measuring the time required for a task, detecting missing operations, and assessing the performance of workers [21].

Aehnelt et al. [1] analyzed factory assembly tasks, and discussed requirements of recognition system for the assembly tasks. Feldhorst et al. [20] recognized activities of manual order picking in a logistics center using an SVM and a random forest. In addition, Rueda et al. [53] proposed a model called CNN-IMU to handle multiple IMU sensor input and evaluated it on order picking activities. Reining et al. [67] used attribute representations to recognize irregular actions in order picking. Attribute representations are atomic human movements and poses, and an activity can be defined as a set of attributes. For example, "walk" can be defined as "left foot or right foot, forward and upright." This representation enables us to easily alter the definition of an activity. However, factory activities are too complex to describe using such attributes. In this study, we use data collected in a real logistics center for evaluation and design a neural network tailored to recognizing ordered work activities using limited training data. Xia et al. [95, 94] leveraged the information extracted from work instruction documents, such as standard duration of an activity and semantic similarity between activities, to recognize factory activities using wrist-worn accelerometers in an unsupervised manner. AI-Amin et al.[3, 2] recognized the assembly process of a three-dimensional printer. They used an accelerometer, a gyroscope, and electromyograms (EMG) as well as a CNN-based model for classification. An application for workers in a meat factory has also been studied [69]. In contrast, this study proposes a neural network-based method for industrial domains that is effective even when the amount of training data is limited.

Although many public datasets of daily activity recognition using IMU sensors are available, there are few public datasets of industrial domains. Stiefmeier et al. [86] collected data of automobile inspection tasks and classified a segment of the sensor data into six classes. Niemann et al. [59] created a dataset called "LARa" for activity recognition of manual order picking and packing tasks conducted in a logistics center. They constructed a simulated experiment environment of a logistics center and collected



(a) Directed Graph of Work Flow (b) Transition Matrix

Figure 2.3: (a) Example flow of work process represented in directed graph. Each node corresponds to activity. Number in node represents identifier of activity. Directed edge shows possible transition between activities. (b) Example transition matrix constructed from example directed graph. Value of (i,j)-th element in matrix shows if edge from i-th to j-th activity exists; 1: True and 0: False.

data from 14 subjects. There are several datasets available for video-based activity recognition in the industrial domains. Dallel et al. [16] constructed a dataset called "InHARD" containing 13 different human activities in an industrial environment. Ben-Shabat et al. [10] created the "IKEA ASM" dataset, which records furniture assembly activities. This dataset consists of sensor data from multi-view RGB cameras and a depth camera.

2.3 Activity Recognition with LOS-Net

2.3.1 Preliminaries

This study assumes that workers wear body-worn inertial sensors, i.e., three-axis accelerometers. In our experiment, the workers wear smartwatches on both their wrists, with the accelerometers of the smartwatches collecting data at a sampling rate of approximately 30 Hz. The workers perform repetitive works in a predefined order of activities, as shown in Fig. 2.1. The goal of this study is to predict the activity class for each time step when a small amount of training data from a target user is given.

As mentioned above, we assume that the order of activities is predefined. Note that

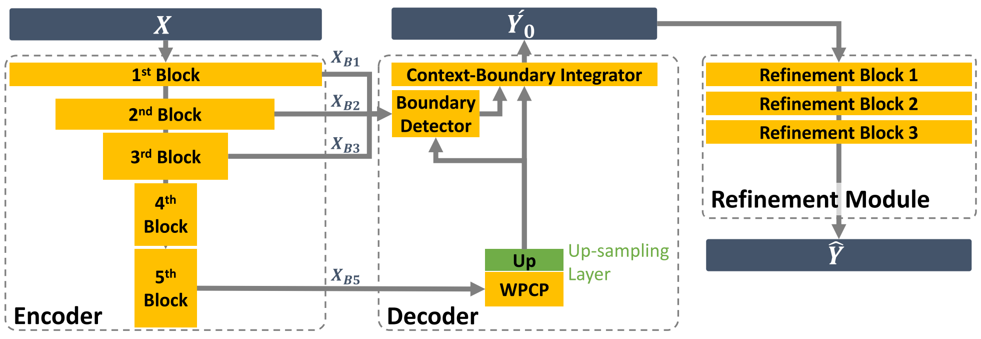


Figure 2.4: Architecture of LOS-Net

although the flow of a work process is a sequence in many cases, a work process can have a structure of a directed graph, as shown in Fig. 2.3 (a). This is because the flow of a work process can change depending on the condition of the item, condition of the work place, and other factors. An example case is that a worker assembles a new box only when the box into which he/she has just packed becomes full.

2.3.2 Overview

The structure of the proposed network (LOS-Net) is shown in Fig. 2.4. The input of the network is a segment of six-axis acceleration data $\boldsymbol{X} \in \mathbb{R}^{6 \times N_T}$ with length N_T as follows:

$$\boldsymbol{X} = [x_1, x_2, ..., x_t, ..., x_{N_T}].$$

The output of the network is a series of activity estimates $\hat{\boldsymbol{Y}} \in \mathbb{R}^{N_C \times N_T}$ with length N_T as follows:

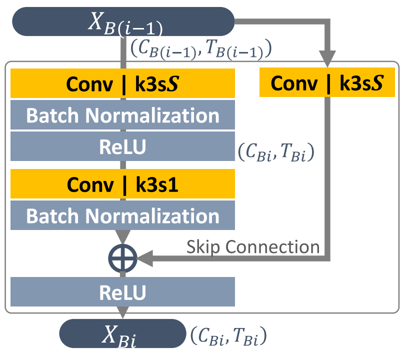
$$\hat{\boldsymbol{Y}} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_t, ..., \hat{y}_{N_T}].$$

For each data point x_t at time t, the network outputs a class probability vector $\hat{y}_t \in \mathbb{R}^{N_C \times 1}$, which is an N_C-dimensional vector, with the c-th element presenting the class probability of the c-th activity class. N_C is the number of activity classes.

As shown in Fig. 2.4, the LOS-Net is composed of three main components: encoder, decoder, and refinement module. The encoder is composed of multiple convolution layers (residual blocks [27]) and extracts features in various time scales (i.e., long- and short-term trends). The decoder processes the extracted features to output the class estimate for each time step. The decoder is composed of three components: Work Process Context Pooling (WPCP) module, Boundary Detector, and Context–Boundary Integrator. The WPCP module extracts the long-term context of the work process (i.e., ordered work) from representations with low temporal resolutions extracted from the last block of the encoder. Because the size of some convolution kernels in the WPCP module is large, the kernels can collect information from distant time steps, which will be introduced in Fig. 2.6 (a). Therefore, the module can recognize information regarding previous/following activities using the kernels. For example, when the center of a kernel is at time t and the time corresponds to the start time of activity C, the kernel with a large size can collect information regarding the activities conducted before activity C. When activities A and B are conducted in this order before activity C, the WPCP module can predict the current activity at time t by referring to this information collected by the kernel. The boundary detector extracts fine-grained information regarding the activity boundaries. The context–boundary integrator fuses the long-term context of the work process and the information on the boundaries to output the class estimate for each time step. The refinement module of the LOS-Net refines the outputs of the decoder based on prior knowledge on the order of activities.

Because the duration of some activities is extremely short, as shown in Fig. 2.1, it is difficult for a window-based approach to handle these activities, as mentioned in Section 2.2.1. Therefore, we decided to use an encoder–decoder architecture, enabling the prediction of a class label for each data point. To achieve a light-weight encoder–decoder model in comparison to normal encoder–decoder models, we included the boundary detector and WPCP module, which were designed to extract minimal important information. In addition we designed the refinement module to deal with sporadic errors in the outputs of the encoder–decoder model. Because the boundary detector estimates the start/end times of each activity class independently, we can regard the boundary detector as not considering the relationship among the activity classes. Instead, the WPCP module captures the temporal relationship among the activity classes using its dilated convolution kernels.

Here, we introduce the three main modules of the LOS-Net (i.e., encoder, decoder, and refinement module).



(a) Architecture

(b) Parameters

Block	Down-sampling (stride)	C_{Bi}	T_{Bi}
2	Yes ($S = 2$)	32	$N_T/2$
3	Yes ($S = 2$)	64	$N_T/4$
4	Yes ($S = 2$)	128	$N_T/8$
5	No ($S = 1$)	256	$N_T/8$

Figure 2.5: Overview of residual block. (a) Architecture of residual block in encoder. "k3s1" denotes parameter setting of convolution layer. For example, it represents kernel of kernel size 3 and stride length 1. "(C_{Bi}, T_{Bi})" presents shape of tensor, i.e., C_{Bi}-dimensional time series with length T_{Bi}. (b) Parameters used in each residual block. C_{Bi} is number of kernels in first and second convolution layers. T_{Bi} is length of output time series.

2.3.3 Encoder

The encoder of the LOS-Net is composed of a convolution block and four residual blocks [27], enabling the construction of a deep model with residual connections. The first convolution block is composed of a 1D-convolution layer with 64 kernels of kernel length 3. The rationale of using only the first block as the convolution layer is that using raw sensor data for a skip connection is irrelevant. Each residual block is mainly composed of two 1D-convolution layers, as shown in Fig. 2.5 (a). In a residual block, the input of the block (processed by one convolution layer for a skip connection) and the output of the second convolution layer are added to avoid gradient diffusion (i.e., skip connection) as follows:

$$\boldsymbol{X}_{Bi} = \mathrm{ReLU}(1\mathrm{DConv2}(\boldsymbol{X}_{B(i-1)}) + 1\mathrm{DConv}(\boldsymbol{X}_{B(i-1)})),$$

where $\boldsymbol{X}_{Bi}$ is the output of the i-th block, 1DConv2() denotes two 1D-convolution layers with batch normalization in the residual block, 1DConv() denotes the 1D-convolution layer for a skip connection, and ReLU() is the rectified linear activation function.

The deeper block in the encoder outputs shorter representations in time (i.e., lower resolution) with a higher dimension, which describe the long-term contexts. The table in Fig. 2.5 (b) lists the number of kernels in the first and second convolution layers in a residual block (C_{Bi}) as well as the length of the output time series (T_{Bi}). The 2nd, 3rd, and 4th blocks in the encoder down-sample the input time series using a kernel stride length of two. For example, the output of the second block is a 64-dimensional time series with length $N_t/2$. The output of the 5th block is a 256-dimensional time series with length $N_t/8$.

2.3.4 Decoder

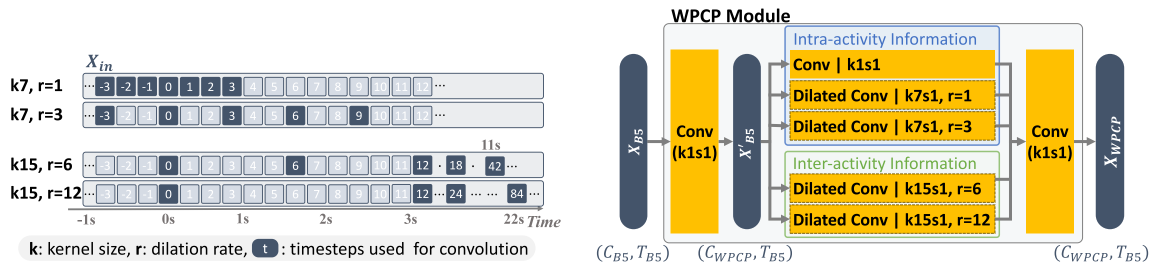
As mentioned above, the decoder is composed of three components: WPCP module, boundary detector, and context–boundary integrator. The context–boundary integrator fuses the outputs of the WPCP module and the boundary detector.

Work Process Context Pooling (WPCP) Module

The WPCP module extracts the long-term context of ordered activities from the outputs of the last residual block in the encoder with few parameters. This module is inspired by the studies on image segmentation by Xi and Chen [93, 15]. Note that unlike in image-based segmentation, it is important for our task to capture long-term contexts regarding a sequence of small actions in the previous/next activity as mentioned in the introduction section.

To collect information from distant time steps with few parameters, the WPCP module leverages dilated convolution kernels [15]. As shown in Fig. 2.6 (a), a dilated convolution performs feature extraction using an inflated kernel by inserting holes between the kernel elements, enabling the collection of information from distant time steps with few kernel parameters. For example, the example second kernel shown in Fig. 2.6 (a), i.e., $r = 3$, has a wide receptive field of nineteen time steps with only seven kernel parameters. As expressed in the following equation, a large receptive field ($2C \cdot r + 1$) can be realized with a few parameters ($2C + 1$) by applying the dilated convolution to the segment centered at the i-th element of an input $\boldsymbol{X}_{in}$.

$$\text{DiConv}(\boldsymbol{X}_{in}, i) = \sum_{k=-C}^{C} x^{in}_{i+r\cdot k} w[k + C],$$



(a) Dilated Convolution (b) WPCP Module

Figure 2.6: Overview of WPCP module. (a) Feature extraction from the input $\boldsymbol{X}_{in}$ with a dilated convolution kernel. Four example kernels are presented. The first is a standard convolution kernel with dilation rate $r = 1$, and the other examples are dilated convolution kernels with $r = 3$, $r = 6$, and $r = 12$, respectively. For example, the second example kernel ($r = 3$) uses values of only 0, ± 3, ± 6, and ± 9 time steps for convolution. (b) Using multiple dilated convolution kernels with different dilation rates to extract long-term work process context.

where x_i^{in} is the i-th element in $\boldsymbol{X}_{in}$, r is the dilation rate, which specifies the spacing between the elements in $\boldsymbol{X}_{in}$, w is an array storing the kernel weights of the dilated convolution kernel, and $2C + 1$ is the kernel size (number of kernel weights).

Figure 2.6 (b) shows the architecture of the WPCP module. The output of the last block of the encoder, $\boldsymbol{X}_{B5}$, is fed into the point-wise convolution layer for dimensionality reduction. We denote the output of this convolution as $\boldsymbol{X'}_{B5}$. Subsequently, the WPCP module applies multiple dilated convolutions with different dilation rates to $\boldsymbol{X'}_{B5}$ to obtain $\boldsymbol{X}_{WPCP}$ as follows:

$$\begin{aligned}\boldsymbol{X}_{WPCP} = \mathrm{PwConv}([&\mathrm{PwConv}(\boldsymbol{X'}_{B5}), \mathrm{Conv}(\boldsymbol{X'}_{B5}, k = 7), \mathrm{DiConv}(\boldsymbol{X'}_{B5}, k = 7, r = 3),\\ &\mathrm{DiConv}(\boldsymbol{X'}_{B5}, k = 15, r = 6), \mathrm{DiConv}(\boldsymbol{X'}_{B5}, k = 15, r = 12),]^{\top})\end{aligned}$$

where $\mathrm{DiConv}(\cdot, k, r)$ is a dilated convolution with dilation rate r and kernel length k, and $\mathrm{PwConv}(\cdot)$ is a point-wise convolution, i.e., its kernel size is one. The first three convolutions (intra-activity information in Fig. 2.6 (b)) use small dilation rates to obtain short-term context, e.g., surrounding small actions.

As mentioned above, it is important for our task to capture long-term information related to a sequence of small actions in the previous/next activity unlike the image

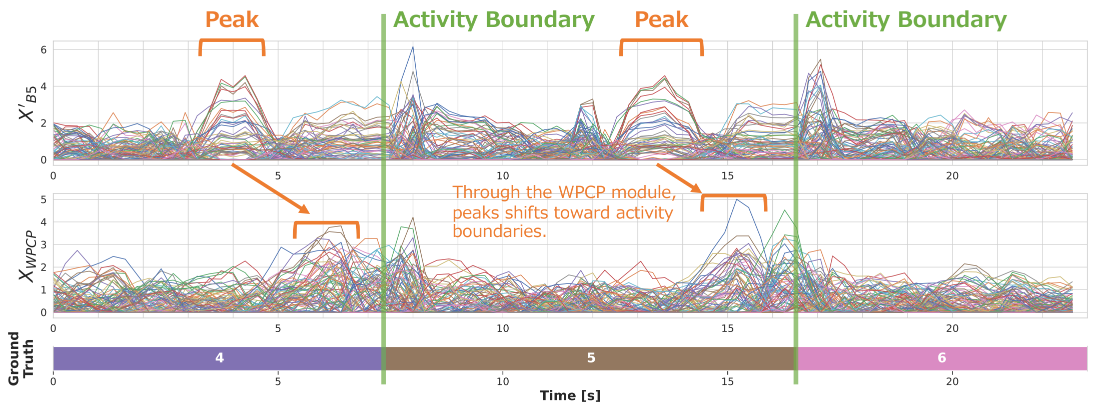


Figure 2.7: Example of $\boldsymbol{X}'_{B5}$ and $\boldsymbol{X}_{WPCP}$ with ground truth activity labels. $\boldsymbol{X}_{WPCP}$ seems to have many peaks around activity boundaries compared to $\boldsymbol{X}'_{B5}$, indicating that dilated convolutions integrate information at distant time steps in $\boldsymbol{X}'_{B5}$.

segmentation task. The main feature of the WPCP module is the extraction of such long-term context through the remaining two convolutions (inter-activity information in Fig. 2.6 (b)) that use larger dilation rates and kernel sizes, enabling information on each of the sequential small actions in the previous/next activity to be convolved. As shown in Fig. 2.6 (a), the dilated convolutions for inter-activity information ($r = 6$ and $r = 12$) can collect information within ranges of up to 11.2 s and 22.4 s[2] before and after the time step of the kernel center, respectively.

Subsequently, linear interpolation is conducted to extend the length of $\boldsymbol{X}_{WPCP}$ to N_T. As above, the WPCP module extracts the long-term context regarding the work process using few parameters (0.4M parameters[3]). Refer to the Evaluation section for the analysis of the number of parameters of the LOS-Net.

Figure 2.7 shows an example output of the WPCP module $\boldsymbol{X}_{WPCP}$ (and $\boldsymbol{X}'_{B5}$) with ground truth activity labels, indicating that the output can approximately capture the trend changes of the sequence of activities. However, we also find that the output cannot precisely present the trend changes (e.g., the boundary between the fifth and sixth activities).

[2] $22.4 = 84 \times 8/30$. Thirty is the sampling frequency of the input signal.

[3] $C_{B5} = 256, C_{WPCP} = 128$. Only the parameters in the convolution layers are considered.

Boundary Detector

The boundary detector extracts fine-grained information regarding the activity boundaries from the output of the encoder. The aim of this module is to provide information on the possible boundaries of *each* activity to the context–boundary integrator. We assume this boundary detection problem to be a multilabel classification problem. As shown in Fig. 2.8, the input of the boundary detector comprises the outputs of the first three blocks of the encoder and the up-sampled outputs of the WPCP module; the output of the boundary detector provides an estimate of the start/end times of each activity. More specifically, the output of the boundary detector is a $2N_C$-dimensional time series with length N_T because the output is a series of estimates of the start/end times of each activity class. For example, a series of estimates of the start times of the c-th activity class is expressed as follows:

$$\hat{\boldsymbol{b}}_c^{st} = [b_1^{st,c}, b_2^{st,c}, ..., b_t^{st,c}, ..., b_{N_T}^{st,c}] \in \mathbb{R}^{1 \times N_T},$$

where $b_t^{st,c}$ is the probability of the start time of the c-th activity at time t. Therefore, the output of the boundary detector is expressed as follows:

$$\hat{\boldsymbol{B}} = \begin{bmatrix} \hat{\boldsymbol{b}}_1^{st}, \\ \hat{\boldsymbol{b}}_1^{ed}, \\ \vdots, \\ \hat{\boldsymbol{b}}_c^{st}, \\ \hat{\boldsymbol{b}}_c^{ed}, \\ \vdots, \\ \hat{\boldsymbol{b}}_{N_C}^{st}, \\ \hat{\boldsymbol{b}}_{N_C}^{ed}, \end{bmatrix} \in \mathbb{R}^{2N_C \times N_T},$$

where $\hat{\boldsymbol{b}}_c^{ed}$ denotes a series of estimates of the end times of the c-th activity.

This study assumes multilabel classification instead of multiclass classification because the latter can mistake the end times with the start times of two consecutive activities, resulting in their miss-detection. Assume that activities A and B are conducted in this order. Because the end times of activity A and start times of activity B are similar in the sensor data, multiclass classification for discriminating them may not work well.

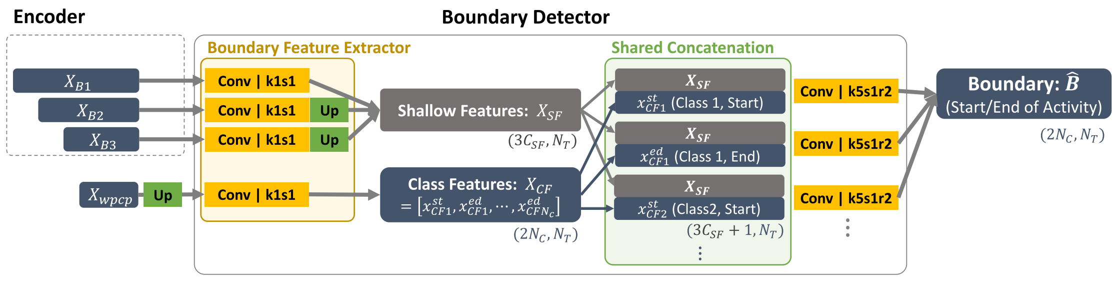


Figure 2.8: Overview of the boundary detector

The architecture of the boundary detector shown in Fig. 2.8 is inspired by the study of Yu et al. [106] on image-based boundary detection. Note that unlike in image-based boundary detection, because the boundaries corresponding to the start and end times of an activity should be different in an activity recognition task, the boundary detector is designed to detect these boundaries individually. We perform boundary detection by leveraging two information types: "shallow features" and "class features." The shallow features are extracted from the shallow blocks in the encoder and contain the information regarding the trend changes with fine temporal resolutions. The class features are extracted from the last block and contain the long-term context information regarding the start/end times of each class. Therefore, the class features have $2N_C$ dimensions (2 corresponds to the start and the end).

To compute the shallow features, as shown in the "Boundary Feature Extractor" of Fig. 2.8, we first apply a point-wise convolution for the dimensionality reduction to the outputs of each block. Following this, we apply linear interpolation to extend the length of the convolved outputs to N_T. The number of dimensions of the outputs from the first, second, and third blocks is reduced to $C_{SF} = 16$. These features are concatenated to form $\boldsymbol{X}_{SF} \in \mathbb{R}^{3C_{SF} \times N_T}$, corresponding to the shallow features.

To compute the class features, we also apply a point-wise convolution to reduce the number of dimensions to $2N_C$ ("Boundary Feature Extractor" in Fig. 2.8). Therefore, the class features, $\boldsymbol{X}_{CF}$, are a $2N_C$-dimensional time series with length N_T. $\boldsymbol{X}_{CF}$

consists of 1D time series with length N_T for the start/end of each class as follows:

$$\boldsymbol{X}_{CF} = \begin{bmatrix} \boldsymbol{x}_{CF1}^{st}, \\ \boldsymbol{x}_{CF1}^{ed}, \\ \vdots, \\ \boldsymbol{x}_{CFc}^{st}, \\ \boldsymbol{x}_{CFc}^{ed}, \\ \vdots, \\ \boldsymbol{x}_{CFN_C}^{st}, \\ \boldsymbol{x}_{CFN_C}^{ed}, \end{bmatrix} \in \mathbb{R}^{2N_C \times N_T},$$

where $\boldsymbol{x}_{CFc}^{st}$ is a time series for the start times of the c-th activity class and contains the long-term context regarding these start times.

Subsequently, we perform multilabel classification by fusing the shallow and class features. For the start/end times of each activity class, first, we concatenate the copied shallow features with the corresponding class features, resulting in a tensor with shape $3C_{SF} + 1 \times N_T$ as shown in "Shared Concatenation" in Fig. 2.8. For example, for the start times of the c-th activity class, we concatenate the shared features $\boldsymbol{X}_{SF}$ with the class features of c-th class $\boldsymbol{x}_{CFc}^{st}$. After this procedure, we obtain the prediction results (e.g., $\hat{\boldsymbol{b}}_c^{st}$) of the start/end times of each class by feeding the concatenated tensor to a convolution layer, enabling to predict $\hat{\boldsymbol{b}}_c^{st}$ using its related features, i.e., $\boldsymbol{X}_{SF}$ and $\boldsymbol{x}_{CFc}^{st}$. Finally, we produce the output of boundary detector $\hat{\boldsymbol{B}}$ by concatenating the prediction results.

To estimate the start/end times of the activity, we employ the Tversky loss function [75] in the model training. The boundary detection task suffers from the class imbalance problem as the number of negative samples overwhelms the number of positive samples. Tversky loss is a loss function that is suitable for learning such imbalanced data by independently adjusting the weight of both precision and recall. The loss function based on the Tversky loss for multilabel classification can be calculated as follows:

$$\mathcal{L}_b = \sum_{c,p \in \{st,ed\}} \mathrm{TL}(\hat{\boldsymbol{b}}_c^p, \boldsymbol{b}_{c,GT}^p, \beta_{fp}, \beta_{fn}),$$

where $\hat{\boldsymbol{b}}_c^p$ denotes the estimates of the start/end times of the c-th activity, and $\boldsymbol{b}_{c,GT}^p$ denotes the ground truth of the start/end times of the c-th activity. Tversky loss $\mathrm{TL}(\cdot)$ is defined as follows:

$$\mathrm{TL}(\hat{\boldsymbol{b}}_c^p \boldsymbol{b}_c^p, \beta_{fp}, \beta_{fn}) = \frac{\sum_{i=1}^{N_T} \hat{b1}_i b1_i}{\sum_{i=1}^{N_T} \hat{b1}_i b1_i + \beta_{fp} \sum_{i=1}^{N_T} \hat{b1}_i b0_i + \beta_{fn} \sum_{i=1}^{N_T} \hat{b0}_i b1_i},$$

where $\hat{b1}_i$ and $\hat{b0}_i$ denote the estimated probabilities for the positive/negative class at time step i, and $b1_i$ and $b0_i$ are the associated ground truths. β_{fp} and β_{fn} are the weights for false positives and false negatives, respectively.

We train the boundary detector to minimize the error between $\hat{\boldsymbol{b}}_c^n$ and $\boldsymbol{b}_{c,GT}^n$ using the above loss function. Note that the ground truth value for $\boldsymbol{b}_{c,GT}^{st}$ at time t is defined as follows:

$$b_{t,GT}^{st,c} = \begin{cases} 1 & \text{if MinTempDist}(S_{st,c}, t) < T_{bd} \\ 0 & \text{otherwise} \end{cases},$$

where $S_{st,c}$ is a set of the start times of the c-th activity class, MinTempDist$(S_{st,c}, t)$ calculates the minimum temporal absolute distance between time t and its closest starting time of the c-th activity, and T_{bd} is the threshold for defining the boundaries. Therefore, the boundary detector is trained to detect $2T_{bd}$-second segments centered at the start/end time. For the model training, refer to Section 2.3.6 in detail.

Figure 2.9 shows an example output of the boundary detector with ground truth activity labels. Although there are several false positives of the detected boundaries (start and end times), the boundaries between consecutive activities are precisely detected.

Context–Boundary Integrator

The context–boundary integrator integrates the outputs of the WPCP module and the boundary detector to predict the activity estimate for each time step, as shown in Fig. 2.10. To utilize the fine-grained boundary information predicted for each activity class, this module first extracts the features for each class and subsequently combines the features of all classes to perform multiclass classification. First, the up-sampled output of the WPCP module and the output of the boundary detector are concatenated by shared concatenation as shown in "Shared Concatenation" in Fig. 2.10. For example, we concatenate up-sampled $\boldsymbol{X}_{WPCP}$, $\hat{\boldsymbol{b}}_c^{st}$, and $\hat{\boldsymbol{b}}_c^{ed}$ for the c-th class. The first convolution layer processes the concatenated tensor for each class. We concatenate the outputs from the first convolution layer for all classes, and feed it to the second, third and fourth convolution layers, enabling the consideration of the relationships among the classes

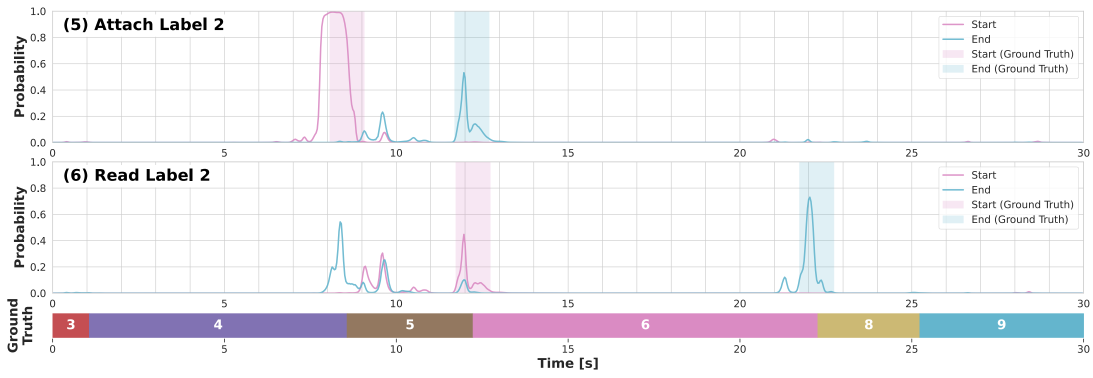


Figure 2.9: Example output of boundary detector with ground truth activity labels. Upper graph shows estimates of start/end times of “Attach Label 2” activity. Lower graph shows estimates of start/end times of “Read Label 2” activity.

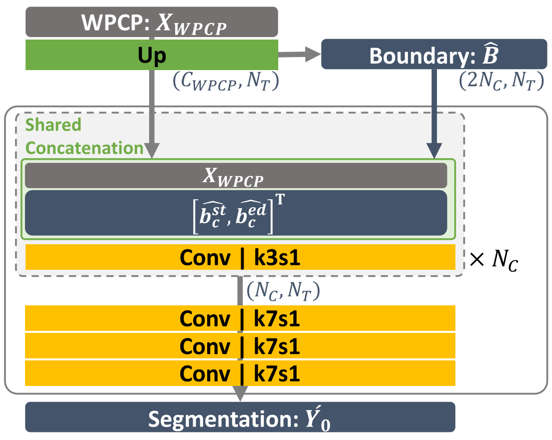


Figure 2.10: Overview of the context–boundary integrator

in generating the multiclass classification results, $\acute{\boldsymbol{Y}}_0$; this is an N_C-dimensional time series with length N_T.

2.3.5 Refinement Module

The refinement module corrects the sporadic errors in the output of the decoder. The input of the module is $\acute{\boldsymbol{Y}}_0$, which is the output of the decoder, and its output is a corrected series of the activity estimates, $\hat{\boldsymbol{Y}}$. The input (output) is an N_C-dimensional time series with length N_T. A N_C-dimension vector at time t in the time series provides the class probability of each class at time t. Fig. 2.11 (a) is an example output of the decoder, showing an N_C-dimensional time series as a heat map. We can see many sporadic errors in the estimates.

The refinement is conducted based on a predefined order of activities. The refinement module corrects the erroneous transitions in $\acute{\boldsymbol{Y}}_0$ that are not specified in the predefined order. In Fig. 2.3 (a), the order (flow) is presented as a directed graph. We convert the graph into a transition matrix $\boldsymbol{M}$, as shown in Fig. 2.3 (b), which is used in the refinement module. $\boldsymbol{M}$ is an $N_C \times N_C$ matrix, with the (i, j)-th element showing whether the transition from the i-th to j-th activity is available in the directed graph. Here, $m_{i,j}$ is the (i, j)-th element of $\boldsymbol{M}$ and is calculated as follows:

$$m_{i,j} = \begin{cases} \mathrm{NaN} & \text{if } i = j \\ 1 & \text{else if the graph has an edge from the } i\text{-th activity to the } j\text{-th activity} \\ 0 & \mathrm{otherwise} \end{cases},$$

We believe that prior information should be simple to the maximum extent because we assume that it is prepared by non-activity recognition researchers. Therefore, this study uses a simple directed graph (and a transition matrix) as the prior information.

Figure 2.12 (a) shows the architecture of the refinement module consisting of three refinement blocks. In each refinement block, which is shown in Fig. 2.12 (b), first we calculate the transition costs, which represent the costs of the activity transitions in the input of the block. When a transition that is not specified in the transition matrix occurs, its corresponding cost is high. Subsequently, we apply convolution layers to correct the estimates near high-transition costs.

Computing Transition Costs

Using $\boldsymbol{M}$ and the input of the refinement block, $\acute{\boldsymbol{Y}}_n$ ($\acute{\boldsymbol{Y}}_0$, $\acute{\boldsymbol{Y}}_1$, or $\acute{\boldsymbol{Y}}_2$), we compute the transition costs, $\boldsymbol{C} \in \mathbb{R}^{1 \times N_T}$, which are a series of length N_T whose t-th element c_t shows a transition cost at time t calculated as follows:

$$\boldsymbol{c}_t = \begin{cases} 0 & \text{if } \arg\max(\acute{y}_t) = \arg\max(\acute{y}_{t+1}) \\ \alpha & \text{else if } m_{i,j} == 1 \\ 1 & \text{else if } m_{i,j} == 0 \end{cases},$$

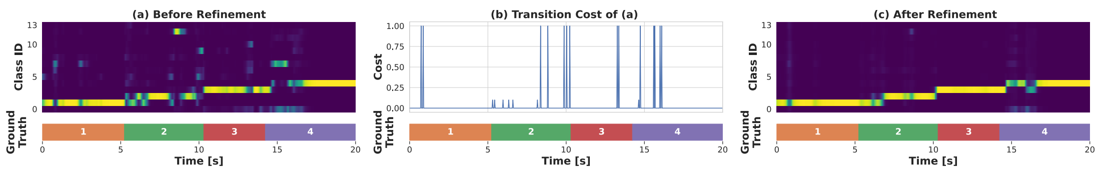
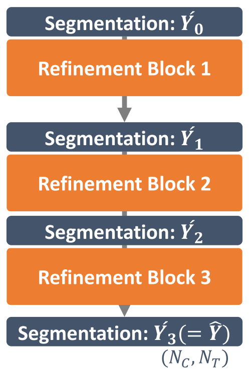
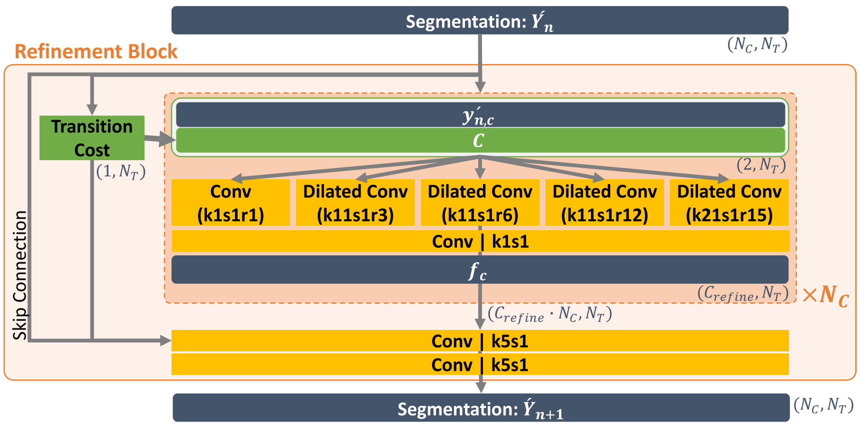


Figure 2.11: Example of estimate correction by refinement module. (a) Example output of decoder $\acute{\boldsymbol{Y}}_0$. (b) Calculated transition costs from decoder output. α is 0.1. (c) Example output of refinement module $\hat{\boldsymbol{Y}}$.


Segmentation: $\acute{Y}_0$
Refinement Block 1
Segmentation: $\acute{Y}_1$
Refinement Block 2
Segmentation: $\acute{Y}_2$
Refinement Block 3
Segmentation: $\acute{Y}_3(=\hat{Y})$
(N_C, N_T)

(a) Refinement Module



(b) Refinement Block

Figure 2.12: Overviews of (a) refinement module and (b) refinement block

where α is the unified cost of the possible transitions $(0 < \alpha < 1)$ and $\acute{y}_t$ is an element of the input $\acute{\boldsymbol{Y}}_n$ ($\acute{\boldsymbol{Y}}_0$, $\acute{\boldsymbol{Y}}_1$ or $\acute{\boldsymbol{Y}}_2$) at time t, i.e., the class estimate at time t. Therefore, when a transition of the class estimates from time t to time $t+1$ occurs and is unavailable in the transition matrix, i.e., the element corresponding to the transition is zero, a transition cost at time t c_t is one. When the transition occurs and is available in the transition matrix, i.e., the element corresponding to the transition is one, c_t is α. As shown in the example in Fig. 2.11 (b), c_t has a large value when sporadic errors occur in the estimates of the decoder. When feeding the transition costs C into the network, the moving sum with window size w_{cost} is applied to smooth out the costs.

Refining Estimates

Each refinement block corrects the errors in its input $\acute{\boldsymbol{Y}}_n$, i.e., the output of the last block, using $\acute{\boldsymbol{Y}}_n$ and transition costs $\boldsymbol{C}$. First, we apply a dilated convolution to correct the prediction errors for each activity class. This is because information from distant time steps is necessary to correct the errors having relatively long durations, e.g., the sporadic error at the beginning of the fourth activity in Fig. 2.11 (a). Subsequently, we combine the results of the above process for all classes to correct the errors by considering the relationships among the classes. Note that, as shown in Fig. 2.12 (b), we introduce a skip connection to prevent ambiguous class estimates at the activity boundaries. In our preliminary experiment, we found that class estimates around an activity boundary become ambiguous, i.e., the probability values corresponding to the correct classes become small, even when the transition is correct. This is because of the above convolution operators, which collect the information within a long time window and convolve them. Therefore, we employ a skip connection to recover the original high probability values in the input of this refinement block $\acute{\boldsymbol{Y}}_n$.

First, for the c-th class, we extract $\acute{\boldsymbol{y}}_{n,c} \in \mathbb{R}^{1\times N_T}$ from $\acute{\boldsymbol{Y}}_{\boldsymbol{n}} \in \mathbb{R}^{N_C\times N_T}$. Here, $\acute{\boldsymbol{y}}_{n,c}$ is a 1D time series with length N_T, whose element at time t denotes the class probability of the c-th class at time t. We concatenate $\acute{\boldsymbol{y}}_{n,c}$ with the transition costs $\boldsymbol{C} \in \mathbb{R}^{1\times N_T}$ and apply point-wise and dilated convolutions with different dilation rates as follows:

$$\boldsymbol{f}_c = \text{PwConv}\left(\text{PwConv}\left(\begin{bmatrix}\acute{\boldsymbol{y}}_{n,c}, \\ \boldsymbol{C},\end{bmatrix}\right) + \sum_{r=3,6,12,15} \text{DiConv}\left(\begin{bmatrix}\acute{\boldsymbol{y}}_{n,c}, \\ \boldsymbol{C},\end{bmatrix}, r\right)\right) \in \mathbb{R}^{1\times N_T}.$$

Subsequently, the outputs $\boldsymbol{f}_c$ of all classes, the transition costs $\boldsymbol{C}$, and the input to this refinement block $\acute{\boldsymbol{Y}}_n$ are concatenated and fed into the two convolution layers to output the corrected estimates, $\acute{\boldsymbol{Y}}_{n+1}$.

$$\acute{\boldsymbol{Y}}_{n+1} = \text{1DConv}\left(\text{1DConv}\left(\begin{bmatrix}\boldsymbol{f}_1, \\ \boldsymbol{f}_2, \\ \vdots, \\ \boldsymbol{f}_{N_C}, \\ \boldsymbol{C}, \\ \acute{\boldsymbol{Y}}_n,\end{bmatrix}\right)\right) \in \mathbb{R}^{N_C\times N_T}.$$

Fig. 2.11 (c) shows an example of the corrected estimates by the refinement module, showing that sporadic errors are corrected.

We train the refinement blocks of the refinement module using the following loss function:

$$\mathcal{L}_r = \gamma_1 \mathrm{CE}(\acute{\boldsymbol{Y}}_1, \boldsymbol{Y}) + \gamma_2 \mathrm{CE}(\acute{\boldsymbol{Y}}_2, \boldsymbol{Y}) + \gamma_3 \mathrm{CE}(\acute{\boldsymbol{Y}}_3, \boldsymbol{Y}),$$

where CE() computes the log–softmax cross-entropy and γ_1, γ_2 and γ_3 are trade-off parameters. The first CE() in the loss function is responsible for the first refinement block and calculates the error between the output of the first refinement block $\acute{\boldsymbol{Y}}_1$ and ground truth $\boldsymbol{Y}$. The second and third CE() in the loss function are responsible for the second and third refinement blocks. By employing CE() for each refinement block, we can efficiently guide the parameter update of the independent blocks. In addition, we use a larger value of γ_n for a deeper block, enabling gradual correction of the errors in the predictions of the decoder.

Data Augmentation for Refinement Module

Because we assume that the amount of training data is limited, it is possible that the refinement module cannot learn various patterns of wrong estimates in $\acute{\boldsymbol{Y}}$. Therefore, we perform data augmentation using $\acute{\boldsymbol{Y}}_0$ (and $\boldsymbol{Y}$) and fine-tune the refinement module on the augmented data independent of the encoder and the decoder. Refer to Section 2.3.6 for the procedure of training in detail.

Here, we introduce the procedure for performing data augmentation in this method. Because an input of the refinement module is simple, i.e., a series of probability vectors $\acute{\boldsymbol{Y}}_0$, we can randomly generate various inputs based on it. We randomly generate $\acute{\boldsymbol{Y}}_r$ from $\acute{\boldsymbol{Y}}_0$ in the training data by (i) changing the duration of each activity and (ii) inserting sporadic errors. To generate various $\acute{\boldsymbol{Y}}_r$, first, we randomly extend/shorten the duration of each segment of an activity in $\acute{\boldsymbol{Y}}_0$ by linear interpolation/down-sampling. For example, when activity A occurs at time t_a and its duration is d_a in $\boldsymbol{Y}$, we shorten/extend the duration by randomly sampling the new duration, d_a^{new}. The new duration is sampled from the normal distribution. The mean and variance of the normal distribution is obtained from ground truth $\boldsymbol{Y}$ in training data.

Following this, we randomly insert error predictions in the series of probability vectors. For each segment of an activity, we assume that the prediction errors follow a

Poisson distribution and persist as an exponential distribution, and randomly generate segments of the error predictions based on the assumption. Assume that an activity A occurs between times t_s and t_e. First, we randomly determine the number of error segments for inserting. Subsequently, for each error segment, we randomly determine the start and end times using the above distributions. The start and end times are selected from a segment between times t_s and t_e so that the error segments do not overlap with each other.

2.3.6 Network Training

We train the LOS-Net by backpropagation based on SGD with momentum by minimizing the following loss function:

$$\mathcal{L}(\theta_e, \theta_d, \theta_r) = \gamma_0 \mathcal{L}_c(\theta_e, \theta_d) + \gamma_{bd} \mathcal{L}_b(\theta_e, \theta_d, \theta_r) + \mathcal{L}_r(\theta_e, \theta_d, \theta_r),$$

where θ_e, θ_d, and θ_r are the network parameters of the encoder, decoder, and refinement module, respectively. γ_0 and γ_{bd} are trade-off parameters. In addition, $\mathcal{L}_c(\cdot)$ is the cross-entropy loss associated with the class prediction of the decoder, i.e. $\mathcal{L}_c = \mathrm{CE}(\acute{\boldsymbol{Y}}_0, \boldsymbol{Y})$.

We train the model in two steps as it is difficult to train three refinement blocks simultaneously. First, we train the encoder, decoder, and the first refinement block for N_{e1} epochs with a learning rate of lr_1, using $\mathcal{L}_c$, $\mathcal{L}_b$, and the first term of $\mathcal{L}_r$. Subsequently, we train the second and third refinement blocks for N_{e2} epochs with lr_2 on the augmented data, with $\mathcal{L}_r$. To accelerate the training, the parameters of the second and third refinement blocks were initialized with the weights of the first refinement block that had already been trained in the first step. In the second step of training, the parameters of the encoder, decoder and the first refinement block are fixed.

2.4 Evaluation

2.4.1 Dataset

We evaluated the LOS-Net using acceleration data collected from 11 workers at actual factories and a logistics center. The data were collected from the smartwatches (Sony SmartWatch3 SWR50) worn on both hands, with a sampling rate of approximately 30

Table 2.1: Dataset overview

Worker	A	B	C	D	E	F	G	H	I	J	K
# of periods	40	40	40	40	40	60	100	30	50	20	70
# of activities (N_C)	12	12	16	16	19	19	14	14	9	10	10
AVG period duration	117s	127s	132s	118s	104s	103s	81s	134s	68s	106s	135s
(SD of duration)	(15s)	(20s)	(41s)	(12s)	(64s)	(17s)	(11s)	(66s)	(27s)	(35s)	(54s)
data duration	78m4s	84m57s	88m33s	78m52s	69m31s	103m58s	136m23s	67m11s	57m7s	35m22s	158m0s
work type	SCREW	SCREW	CHECK	CHECK	SCREW	SCREW	CHECK	CHECK	PACK	PACK	PACK

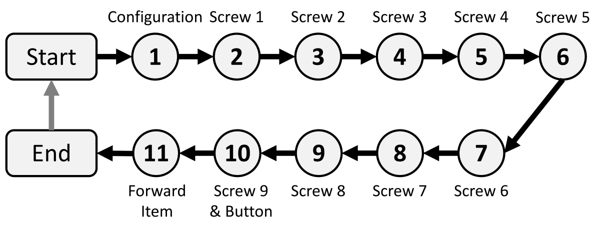
Table 2.2: Work overview

Work Type	Location	Description
SCREW	Factory	Install screws on products
CHECK	Factory	Check final products and record results
PACK	Logistics Center	Pack multiple items in a box

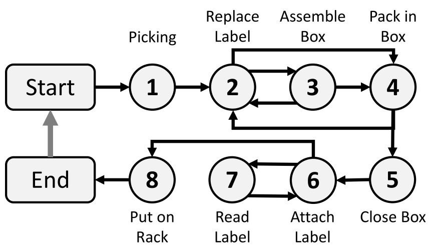
Hz. The collected data were manually annotated to generate the ground truth labels using video recordings. Activity labels were labeled by employed annotators with supervision by industrial engineers. The definitions of the activities were determined based on instruction documents of these tasks prepared by managers of the sites. Table 2.1 provides an overview of our dataset. Note that a “period” in the table implies one iteration of the activities, as shown in the upper part of Fig. 2.1. The work processes conducted by the different workers are different. The “work type” row shows a rough category of the work by each worker. An overview of the work types is provided in Table 2.2. The packing works are highly variable compared to the other works because the size and number of items to pack are different in different periods.

Because a flow of activities in an instruction document of a work process is provided as a tree or a series of ordered activities (with branches described in text) in an Excel file, we constructed a directed graph based on the flow. Fig. 2.13 shows the directed graphs of example work processes.

As a preprocessing, acceleration values greater than $+3$ G or smaller than -3 G were clipped to address the measurement errors. Subsequently, normalization was performed for each axis.



(a) Work Flow of Subject A



(b) Work Flow of Subject I

Figure 2.13: Examples of work flows

2.4.2 Evaluation Methodologies

This study assumes that a limited amount of training data is provided. Therefore, for each worker, first, we divided the obtained periods into sets of periods, with each set being composed of ten periods, resulting in N_{fold} sets. Subsequently, we trained the recognition model on a set of periods and tested the model on the remaining sets (i.e., $N_{fold} - 1$ sets). We repeated this procedure so that each set became the training data once. For robust evaluation, we repeated this cross-validation five times with different random seed settings. The recognition method was evaluated using the macro-averaged F1-measure based on an estimate for each time step. The macro-averaged F1-measure is a widely used metric that is less affected by a class imbalance because it is computed by simply averaging the F1-measures computed for each class. Because our dataset is class-imbalanced, we use this metric for this experiment.

To evaluate the effectiveness of the LOS-Net, we designed the following methods:

- **SVM(W)**: This is a classic machine learning method using a sliding time window. In this study, we used a sliding time window with length 3 seconds without overlap. Handcrafted features were extracted within a window and concatenated to form a feature vector. Specifically, we extracted 14 types of features from each axis based on prior activity recognition studies [70]. SVM with radial basis function (RBF) was used.

- **CNN(W)**: This is a re-implementation of BaselineCNN proposed in a related study [63]. This model is a baseline for deep learning using a sliding time window. Acceleration data within each window were fed into four convolutional layers, followed by two fully connected layers. The number of units and kernel size are set as in the previous study.

- **CNN**: This method performs dense labeling (i.e., predicting a class label for each data point) using only convolutional operations. The network is composed of eight convolutional layers with a kernel length of five, followed by the output layer. The numbers of filters in the first six layers and the last two layers are 64 and 128, respectively.

- **ConvLSTM**: We re-implemented the ConvLSTM model proposed in [63]. The model consists of five convolution layers with stride 1 followed by two LSTM layers with 128 hidden units. The number of units and kernel size were set as in the related studies [63, 109].

- **LOS-Net(+attn32)**: This is a variant of the proposed method, and applies self-attention [89] instead of the WPCP module. The output of the encoder is fed into an attention layer with 32 attention heads. The attention output is then multiplied by the output of the encoder, and then fed into the context–boundary integrator. Because of the slow convergence of this architecture, we trained this model for 800 epochs, which is twice the number of epochs of the proposed method.

- **U-Net**: We re-implemented the model proposed in [109]. This model consists of five encoding and decoding blocks. Each block is composed of two convolution and pooling/unpooling layers. The number of units and the kernel size were set as in the related study [109]. This method is the one of the state-of-the-art models for human activity segmentation.

- **LOS-Net**: This is the proposed method.

In the window-based approaches, i.e., SVM(W) and CNN(W), the prediction result for a window is expanded to the size of the window, enabling calculation of the F1-measure based on all time steps. In the training of window-based models, the activity class at the last time step in the window is used as the ground truth for that window and models are trained to predict them. In the segmentation approaches, a segment of length 60 s (i.e., N_T corresponds to the length of 60-second data) is fed to the models.

In addition, we developed the following methods to evaluate each function in the LOS-Net:

- **LOS-Net(-WPCP)**: This is a variant of the LOS-Net. This method does not use the WPCP module (and the refinement module). Therefore, the output of the fifth block is directly fed to the context–boundary integrator.
- **LOS-Net(-B)**: This is a variant of the LOS-Net. This method does not use the boundary detector (and the refinement module). Therefore, an input of the context–boundary integrator is only an output of the WPCP module.
- **LOS-Net(-R)**: This is a variant of the LOS-Net. This method does not use the refinement module. Therefore, an output of this method is identical to an output of the decoder.
- **LOS-Net(-R2)**: This is a variant of the LOS-Net. This method does not execute the second training step, i.e., refinement using the augmented data.

We implemented the models using PyTorch and Sklearn. We executed this experiment on a GPU cluster. The accelerator used in this experiment is Geforce GTX 1080, Geforce RTX 2070, Quadro RTX 8000, or Titan RTX. Table 2.3 lists the experimental parameters used in this experiment. Different N_{e1} were used depending on the experimental settings and model characteristics. In Zhang's experiment [109], the networks were trained for 100 epochs. However, the number of batches per epoch, i.e., the number of model updates, is smaller in our experiment because the number of samples is smaller. To compensate for this, we increased the number of epochs to $N_{e1} = 300$. For the methods that include boundary detection, i.e., LOS-Net, LOS-Net(-WPCP), and LOS-Net(-R), we set $N_{e1} = 400$. This is because the boundary detection task is harder and requires more iterations to reduce the loss for boundary detection compared to the segmentation task.

Table 2.4 shows the number of parameters for each method. The parameters for the LOS-Net are approximately 1/10 of those for the U-Net. Compared to the U-Net, the number of parameters of the LOS-Net is smaller for both the decoder and encoder. As the LOS-Net resolves long-term dependencies with the WPCP module, we used an encoder with a smaller number of blocks than for the U-Net. Note that to make the

Table 2.3: Experimental parameters

Model		Training	(1st Step)	Refinement	(2nd Step)
C_{WPCP}	128	Optimizer	SGD with momentum	Optimizer	SGD with momentum
C_{SF}	16	– momentum	0.9	– momentum	0.9
C_{refine}	$2N_C$	– lr_1	5.00E-02	– lr_2	1.00E-02
α (transition cost)	0.1	batch size	16	batch size	16
w_{cost}	30pt	N_{e1}	300	N_{e2}	100
		N_{e1} (w/ boundary detection)	400	γ_2	0.95
		γ_0	0.95	γ_3	1
		γ_1	1		
		T_{bd}	15pt		
		β_{fp}	0.4		
		β_{fn}	0.6		

Table 2.4: Number of parameters in each network. The parameters of the convolutional layers, LSTM layers, and fully connected layers were counted. N_T was set to 1800 (= 60 s).

Model	# of Params	Model	# of Params	Model	# of Params
CNN-WIN	7,454,346	U-Net	11,537,162.00	LOS-Net	963,620
CNN	229,130	– Encoder	6,296,096	– Encoder	437,824
DeepConvLSTM	296,330	– Decoder	5,241,066	– WPCP	442,368
				– Boundary Detector	9,508
				– Context–Boundary Integrator	10,500
				– Refinement Module (single block)	21,140

U-Net capture long-term dependencies, it is necessary to use a large number of blocks, resulting in a large number of parameters.

2.5 Results

2.5.1 Recognition Accuracy

Table 2.5 shows the macro-averaged F1-measure for each method, for each worker. Surprisingly, the LOS-Net greatly outperformed the other methods for all the workers. Specifically, the F1-measures of the LOS-Net were much higher than those of the

Table 2.5: F1-measure (and standard deviation shown below F1-measure) of the methods for eleven workers trained on 10 periods of data.

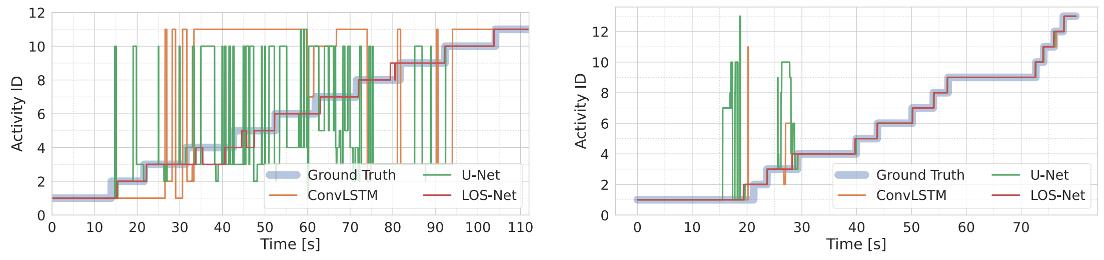
Worker	Work Type	SVM(W)	CNN(W)	CNN	ConvLSTM	U-Net	LOS-Net(+attn32)	LOS-Net
A	SCREW	0.298	0.230	0.220	0.421	0.497	0.494	**0.831**
		(0.005)	(0.028)	(0.022)	(0.055)	(0.055)	(0.017)	(0.029)
B	SCREW	0.335	0.230	0.230	0.377	0.631	0.493	**0.835**
		(0.006)	(0.019)	(0.015)	(0.077)	(0.022)	(0.035)	(0.012)
C	CHECK	0.672	0.590	0.766	0.726	0.852	0.820	**0.880**
		(0.004)	(0.011)	(0.019)	(0.061)	(0.015)	(0.004)	(0.007)
D	CHECK	0.627	0.541	0.775	0.711	0.833	0.811	**0.858**
		(0.006)	(0.037)	(0.017)	(0.123)	(0.029)	(0.006)	(0.008)
E	SCREW	0.250	0.174	0.427	0.547	0.888	0.787	**0.923**
		(0.013)	(0.039)	(0.075)	(0.199)	(0.011)	(0.010)	(0.006)
F	SCREW	0.231	0.211	0.252	0.310	0.727	0.518	**0.795**
		(0.006)	(0.034)	(0.026)	(0.093)	(0.04)	(0.009)	(0.005)
G	CHECK	0.424	0.421	0.752	0.716	0.815	0.743	**0.859**
		(0.007)	(0.009)	(0.015)	(0.055)	(0.034)	(0.011)	(0.009)
H	CHECK	0.249	0.326	0.501	0.425	0.571	0.499	**0.617**
		(0.007)	(0.017)	(0.008)	(0.064)	(0.026)	(0.020)	(0.017)
I	PACK	0.299	0.272	0.431	0.298	0.537	0.369	**0.565**
		(0.009)	(0.016)	(0.02)	(0.051)	(0.011)	(0.009)	(0.009)
J	PACK	0.297	0.227	0.321	0.224	0.398	0.384	**0.505**
		(0.017)	(0.043)	(0.015)	(0.027)	(0.086)	(0.018)	(0.017)
K	PACK	0.318	0.277	0.335	0.282	0.450	0.398	**0.535**
		(0.004)	(0.008)	(0.01)	(0.048)	(0.024)	(0.005)	(0.005)

window-based methods, i.e., SVM(W) and CNN(W). The duration of each activity varied, as shown in Fig. 2.1, resulting in poor performance of the window-based methods. Interestingly, the performance of the SVM(W), which is based on classic machine learning, is not very different from that of the CNN(W). This could be because the CNN(W), which is a feature learning approach, could not extract meaningful features from a limited amount of training data. The CNN outperformed CNN(W) for many workers, indicating the effectiveness of dense labeling in this task. However, the F1-measures of the CNN were still poorer than those of the LOS-Net.

The LOS-Net also significantly outperformed the state-of-the-art recurrent model, i.e., ConvLSTM. Surprisingly, the F1-measure of the LOS-Net was higher than that of the ConvLSTM by more than 30% for four workers. For the recognition of ordered work, the temporal relationship with the previous/next activities is crucial. It seems that the ConvLSTM fails to capture such a long-term context for ordered work sequences with limited training data. Specifically, it was difficult for the LSTM layers in the ConvLSTM to capture the information of the next activity. The LOS-Net also significantly outperformed the state-of-the-art U-Net by about 10% on average even though the number of parameters of the LOS-Net is approximately 1/10 of that of the U-Net. Specifically, the LOS-Net outperformed the U-Net by 3.4–30% for the screwing tasks. The screwing tasks contain similar activities (e.g., screwing different parts), making it difficult to discriminate between these activities. However, the LOS-Net can recognize these activities while considering the long-term context and predefined order of activities. Although an attention mechanism is effective for leveraging sensor data segments that are useful for activity recognition, the training of an attention model is a data-hungry process. Therefore, LOS-Net(+attn32) did not work well under our scenario. Moreover, the F1-measures for the packaging tasks were poorer than those for many other tasks across all the methods. This could be due to the packaging tasks being highly variable as the number of items to pack varies in different periods. However, the LOS-Net achieved the highest F1-measures for the packaging tasks.

Table 2.5 also shows the standard deviations of the F1-measures (inside parentheses). The standard deviations of the state-of-the-art deep models (U-Net and ConvLSTM) are large in many cases, as shown in Fig. 2.2. In contrast, the standard deviations of the estimates for the classic model, i.e., SVM(W), and the LOS-Net are small. As the SVM(W) uses hand-crafted features, the estimates of the method are stable. In contrast, the U-Net and ConvLSTM do not seem to learn effective features with limited training data. These results also indicate that the LOS-Net can learn efficient features with limited training data.

Figure 2.14 shows example estimates of the LOS-Net, U-Net, and ConvLSTM. The LOS-Net can precisely recognize activities with small errors at several activity boundaries. In the case of worker A, activities 2–10 comprise screwing activities. These activities were incorrectly estimated as activities 3, 10 or 11 by the U-Net and ConvLSTM as shown in Fig. 2.14 (a), resulting in the low macro-averaged F1-measures of these methods. In the screwing work, similar screwing activities are iterated several times; therefore, it is important to capture long-term information on surrounding activities to



(a) Worker A (b) Worker G

Figure 2.14: Examples of ConvLSTM, U-Net, and LOS-Net outputs

precisely recognize these activities. U-Net and ConvLSTM seem to fail to capture these long-term dependencies, resulting in incorrect classification of all screwing activities into specific classes (i.e., activities 3, 10, and 11). Similar incorrect predictions were observed for workers B, E, F, I, J, and K.

The hand motions of activities in the item checking work are dis-similar with each other unlike the screwing work. Therefore, even when long-term dependencies cannot be extracted, these activities can be recognized with relatively high accuracy. However, U-Net and ConvLSTM sometimes made wrong predictions when the duration of the activity was long such as activity 1 in Fig. 2.14 (b). The LOS-Net suppresses such errors by leveraging prior knowledge on the order of the work process.

2.5.2 Contributions of WPCP module, Boundary Detector, and Refinement Module

As shown in Table 2.6, the F1-measures of LOS-Net(-WPCP) are much poorer than those of LOS-Net, suggesting that the WPCP module is the best contributor. Therefore, capturing long-term context is necessary to achieve the high recognition performance for the ordered works. In addition, as shown through the results of LOS-Net(-R) and LOS-Net, the refinement module also improves the F1-measures by approximately 2–5% for many workers. The refinement module also leverages the predefined order of activities, indicating that information on the order of activities is crucial in this task. The results of LOS-Net(-R2) and LOS-Net suggest that the data augmentation conducted for

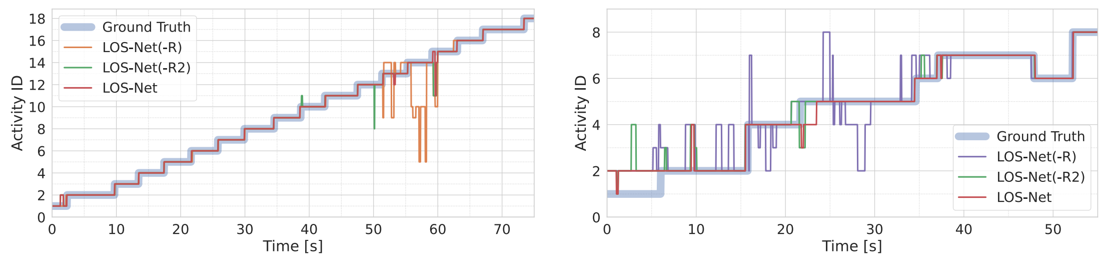
Table 2.6: Results of ablation study

		w/o Refinement Module			w/ Refinement Module	
Worker	Work Type	LOS-Net(-WPCP)	LOS-Net(-B)	LOS-Net(-R)	LOS-Net(-R2)	LOS-Net
A	SCREW	0.372	0.804	0.750	0.811	**0.831**
		(0.018)	(0.015)	(0.043)	(0.025)	(0.029)
B	SCREW	0.366	0.828	0.790	0.801	**0.835**
		(0.021)	(0.026)	(0.048)	(0.046)	(0.012)
C	CHECK	0.805	0.875	0.871	0.874	**0.880**
		(0.008)	(0.003)	(0.005)	(0.004)	(0.007)
D	CHECK	0.815	0.857	0.854	0.855	**0.858**
		(0.004)	(0.005)	(0.004)	(0.006)	(0.008)
E	SCREW	0.729	0.901	0.902	0.917	**0.923**
		(0.011)	(0.008)	(0.018)	(0.006)	(0.006)
F	SCREW	0.481	0.743	0.771	0.781	**0.795**
		(0.008)	(0.016)	(0.003)	(0.006)	(0.005)
G	CHECK	0.754	0.734	0.856	0.855	**0.859**
		(0.018)	(0.201)	(0.019)	(0.013)	(0.009)
H	CHECK	0.506	0.541	0.599	0.608	**0.617**
		(0.014)	(0.146)	(0.019)	(0.016)	(0.017)
I	PACK	0.381	0.511	0.534	0.551	**0.565**
		(0.01)	(0.017)	(0.010)	(0.006)	(0.009)
J	PACK	0.329	0.493	0.489	0.489	**0.505**
		(0.01)	(0.008)	(0.016)	(0.013)	(0.017)
K	PACK	0.327	0.532	0.518	0.528	**0.535**
		(0.007)	(0.005)	(0.013)	(0.007)	(0.005)

the refinement also improved the F1-measures by approximately 1–2% in many cases. Moreover, while it seems to be difficult to recognize the ordered work using only the boundary detector, the boundary detector can fine-tune the prediction results.

2.5.3 Effect of Refinement

Figure 2.15 shows example estimates with and without the use of the refinement module; LOS-Net(-R), LOS-Net(-R2), and LOS-Net. The prediction of the model without the refinement module, i.e. LOS-Net(-R), contained several sporadic errors as shown in Fig. 2.15 (a). Activity 5 of worker I is a long-duration activity, lasting more than 10 s,



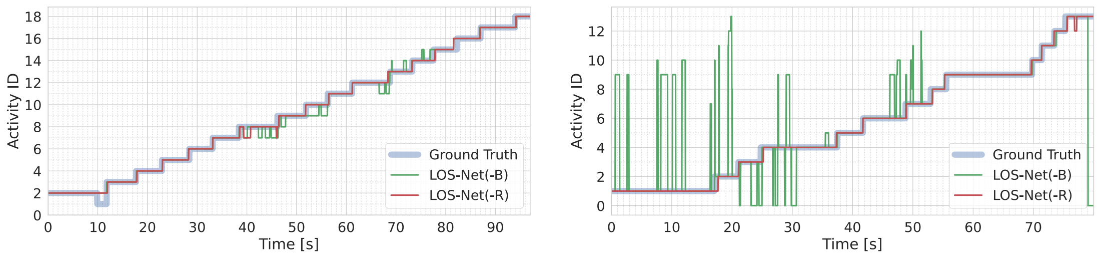
(a) Worker E (b) Worker I

Figure 2.15: Examples of LOS-Net(-R), LOS-Net(-R2), and LOS-Net outputs

and the workflow shown in Fig. 2.13 (b) shows that there are no transitions from activity 5 to activities 2, 7 and 8. However, the results of LOSNet(-R) contained transitions from activity 5 to activities 2, 7, and 8 multiple times, as shown in Fig. 2.15 (b). The methods using the refinement module, i.e., LOS-Net and LOS-Net(-R2), managed to suppress these incorrect transitions.

2.5.4 Effect of Boundary Detection

The boundary detector greatly improved the F1-measures for workers G and H as shown in Table 2.6. Fig. 2.16 indicates that the boundary detector has an ability to suppress sporadic errors. In contrast, the boundary detector was not very effective for workers A–E. This can be because we could achieve very high F-measures with only the WPCP module. As for workers I–K, the boundary detector was also not very effective. Table 2.7 shows average precision (AP) and its standard deviation of the boundary detection in the LOS-Net. The AP was first calculated for each class and then averaged across all classes. The APs of the boundary prediction were poor for workers I–K. This can be because the tasks undertaken by them, i.e., packing works, were highly variable. This is the reason why the boundary detector was not effective for these workers.



(a) Worker F (b) Worker G

Figure 2.16: Examples of LOS-Net(-B) and LOS-Net(-R) outputs

Table 2.7: Classification accuracy of boundary detection (macro-averaged precision)

Worker	A	B	C	D	E	F	G	H	I	J	K
AP	0.485	0.554	0.682	0.722	0.790	0.655	0.662	0.376	0.172	0.072	0.089
(SD)	(0.043)	(0.055)	(0.011)	(0.011)	(0.046)	(0.016)	(0.014)	(0.006)	(0.021)	(0.014)	(0.004)

2.5.5 Discussion

Amount of Training Data

The experimental evaluation used 10 periods of data as training data, which comprised data with a duration of approximately 15 minutes. Here, we investigate the effect of the amount of training data on the recognition performance of the methods. We used a set of ten-period data as test data and the remaining periods were used as a pool of training data. Fig. 2.17 shows the F1-measures of the U-Net and LOS-Net when we changed the amount of training data. As shown in the results, the LOS-Net outperformed the U-Net in many cases even when the amount of training data is large. This can be because the WPCP module, which has a large receptive field, captures long-term contexts that are difficult to capture by the U-Net. As for workers A and B, the LOS-Net greatly outperformed the U-Net when the number of training period was five. Even when the number of training period is less than five, the LOS-Net outperformed U-Net in many cases. Because the number of parameters of LOS-Net is much fewer than that of U-Net, LOS-Net can efficiently capture crucial information on ordered work with limited training data. Specifically, dilated convolution kernels of the WPCP module provide

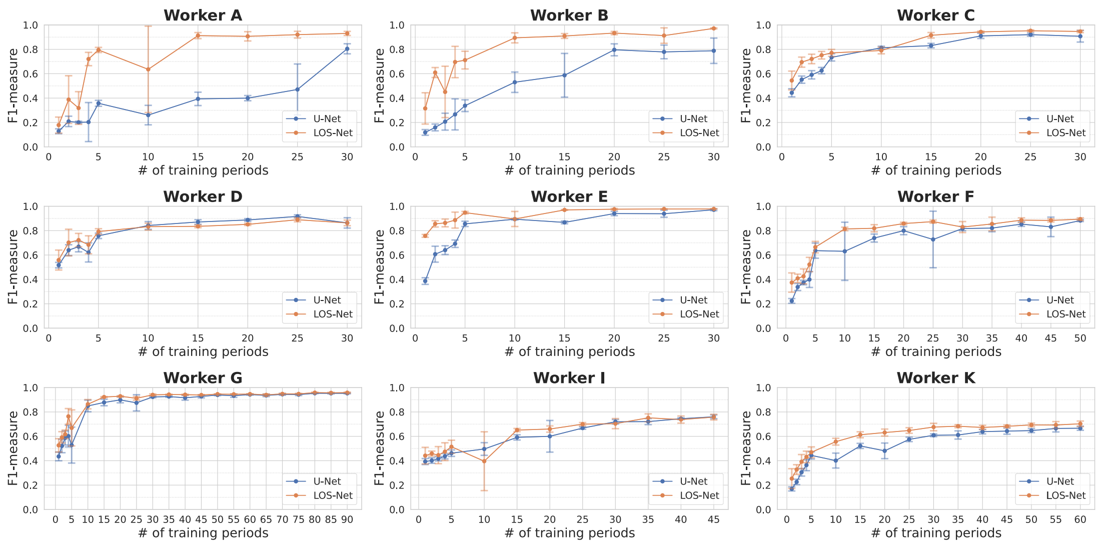


Figure 2.17: Transitions of recognition accuracy of U-Net and LOS-Net when we vary the amount of training data. This experiment was conducted five times by changing the random seed. Because the amount of data is limited for workers H and J, we did not conduct this experiment for these workers.

a wide receptive field with few parameters. Note that, when the number of training periods is low (e.g., 1–5), the test accuracy of the methods seems to be unstable. This might be because when the amount of training data is limited, the test accuracy is greatly affected by the quality of the limited training data applied. For example, sensor data of packing tasks greatly vary depending on items to pack. When outlying periods, e.g., packing many items or packing vary large items, are included in training periods, these outlying periods have a negative impact on the training process of LOS-Net. However, LOS-Net still outperforms U-Net even when the number of training periods is very low, indicating the effectiveness of LOS-Net in such cases.

2.6 Conclusion

This study presented a new activity recognition method for ordered manual tasks, such as assembly work and packaging tasks in factories. The proposed LOS-Net is a lightweight model that can be trained on a limited amount of training data. The LOS-Net is designed to extract necessary information related to the order of work: (i) the boundary information between consecutive activities and (ii) the long-term context regarding the ordered works, e.g., information regarding the previous/next activities. The LOS-Net also has a module for correcting sporadic errors in estimates according to prior knowledge related to the predefined order of activities. Using data collected in actual industrial scenarios, our experiment demonstrated the effectiveness of the LOS-Net. The LOS-Net significantly outperformed other state-of-the-art deep models. As a part of our future study, we plan to extend our method to solve the panoptic segmentation task [34].

Chapter 3

OpenPack: A Large-scale Dataset for Recognizing Packaging Works in IoT-enabled Logistic Environments

In this chapter, we describe our new large-scale public dataset in industrial domain. We also explore method to fuse sensor and IoT data in efficient way.

3.1 Introduction

In industrial environments such as factories and logistics centers, human workers continue to perform important roles in ensuring flexible responses to rapidly changing demands of customers and suppliers. Specifically, the number of workers employed in logistics continues to increase [49, 77], and this trend is expected to persist in the future [36, 66, 100]. For example, Amazon ships 2.5 billion packages annually in the U.S., which highlights the significance of streamlining shipping processes in global and regional supply chains. Aiming to streamline work processes performed by human workers and enable enterprises to make business decisions based on granular work information, digitization is being widely implemented in industrial environments as part of the Industry 4.0 transformation. In Industry 4.0, digitized data on human motion acquired from sensor systems and readings from IoT-enabled devices (e.g., connected handheld terminals) are expected to be used to achieve work activity recognition to quantify and streamline work processes performed by human workers. Therefore, activity recognition for human workers in industrial domains has recently attracted attention as a topic of active research [95, 94, 52, 103].

However, the following challenges should be addressed to enhance work activity recognition studies.

- **Lack of datasets for industrial domains**: The amount of publicly available datasets for industrial domains containing complex activities (e.g., manufacturing and packing/picking tasks) remains limited, and many of the available activity recognition datasets focus only on simple daily activities (e.g., walking and running). Fig. 3.1 shows a typical series of packaging tasks iterated several times, with each iteration of the process (i.e., period) comprising a sequence of operations in which the acceleration data indicate the complexity of operations. While a dataset is available for the logistics domain [59], the activity classes used do not reflect the complex work process[1]. In addition, the size of many existing datasets does not suffice to fulfill the basic requirements of deep learning.
- **Limited modality**: Many public datasets for manual tasks provide only vision-related modalities. However, because many types of manufacturing equipment and storage systems are installed in industrial environments, occlusion tends to pose challenges in applying vision-only approaches.
- **Difficulties in human–IoT data coordination**: Although digitization is progressing in actual industrial domains as part of the development of Industry 4.0, to the best of our knowledge, no datasets on activity recognition that include both sensory data on human motions and readings from IoT-enabled devices operated by the human workers are publicly available.
- **Lack of rich metadata**: Many of the available datasets do not provide a rich set of metadata related to manual works such as a set of the items to be packed, which

[1]The LARa dataset [59] does not consider the difference between work activities (e.g., packing items and making a shipping box) in the definition of activity classes. Instead, these activities are defined as handling-related classes, including handling (upward), handling (centered), and handling (downward), which are defined as handling items and boxes at different positions. For example, in handling (upward), a subject handles items with at least one hand reaching shoulder height.

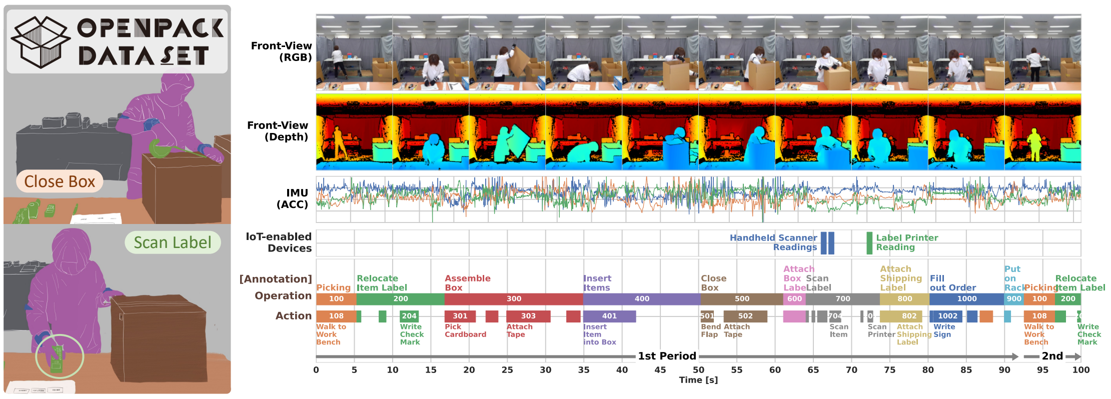


Figure 3.1: Illustration of the OpenPack dataset. A subject iterated a typical series of packaging works, with each iteration of the process (i.e., period) comprising a sequence of complex operations.

limits the understanding of recognition results and the design of new, enhanced research tasks.

To address these challenges, in this study, we propose a new multimodal dataset for packaging work recognition in logistics, called OpenPack (Fig. 3.1). OpenPack consists of $20,129$ instances of activities (operations) and $52,529$ instances of actions with 9 types of modalities captured from 16 distinct subjects with various levels of experience performing packaging tasks. The features of OpenPack are summarized as follows. (i) OpenPack is the first large-scale dataset for packaging work recognition that contains readings from IoT-enabled devices. (ii) OpenPack provides multimodal work activity data, including sensory data from body-worn inertial measurement units (IMUs), depth images, and LiDAR point clouds for use in research on multi-/crossmodal, IMU-only, and vision-only work activity recognition according to conditions in an expected target environment. (iii) OpenPack also provides a rich set of metadata such as subjects' levels of experience in packaging work as well as their physical characteristics, enabling the design of various research tasks (e.g., assessment of workers' performance) in addition to basic work activity recognition.

One of the main features of OpenPack is that it contains readings from IoT-enabled devices operated by subjects. Although high performance is required of work activity recognition methods in industrial domains to develop critical applications such as outlier detection, even state-of-the-art activity recognition methods cannot achieve sufficient performance by using noisy sensor data (e.g., vision-based and IMU sensor data). Leveraging high-confidence readings from IoT-enabled devices, which strongly relate to activities performed (e.g., readings from a handheld scanner to detect the "scan label" operation), is expected to provide a key enabler to achieve precise recognition for real applications.

Although readings from IoT-enabled devices are highly confident, they are sparse in time because in general they are recorded only when a worker operates a device. Therefore, an efficient strategy of activity recognition using such IoT data with noisy sensor data (e.g., IMU data) may consider (i) leveraging available readings from IoT-enabled devices as high-confidence *anchors* and then (ii) predicting an activity class at a time step when readings are unavailable by referring to the sensor data and anchors surrounding the time step of interest. That is, because activity classes can be precisely predicted at a time step when readings from IoT-enabled devices are available, we predict an activity class at a time step when IoT data are unavailable by referring to surrounding IoT data as anchors. However, even for state-of-the-art human activity recognition methods that assume sensor data as input, there is no guarantee that a recognition model trained by these methods would learn the abovementioned efficient recognition strategy if sensor and IoT data were simply fed into the recognition model. Therefore, in this thesis, we propose a novel activity recognition model for sensor and IoT data, called a ladder-shaped two-stream network (LTS-Net), which is designed to be guided to learn the abovementioned strategy. LTS-Net processes sensor data and high-confidence IoT data in different streams and updates information in each stream by referring to information in different streams while preserving information on the IoT data in deeper layers.

The key contributions of this research are summarized as follows: i) To the best of our knowledge, OpenPack is the largest dataset on industrial work activity recognition. We expect this data to contribute to research on industrial AI systems by providing a challenging task, that is, to complex activity recognition via sensor and sparse IoT data. ii) We propose LTS-Net to effectively utilize sparse readings from IoT devices via a ladder structure. iii) The results of the experimental evaluation showed that LTS-Net significantly improved the performance, indicating the usefulness of the readings from IoT devices and the effectiveness of the ladder structure in our method.

Table 3.1: Overview of public datasets of human activities/works. D: Depth, Acc: Acceleration data, Gyro: Gyroscopic data, Ori: Orientation sensor data, EDA: Electrodermal activity, BVP: Blood volume pulse, Temp: Temperature.

Domain	**Datasets**	**Type of Task**	**Recording Length**	**Activity Class***	**# of work period**	**Annotated Instances**	**Subjects**	**Modality**	**IoT Devices**	**View**	**Year**
Multi-modal	MHAD	Daily Actions	82m	11	N/A	660	12	RGB+D+Keypoints+Acc+Mic	No	12	2013
	UTD-MHAD	Daily Actions	9m	27	N/A	180	8	RGB+D+Keypoints+Acc+Gyro	No	1	2015
	MMAct	Daily Actions	17h35m	37	N/A	36,764	20	RGB+D+Keypoints+Acc+Gyro +Ori+WiFi+Pressure	No	4+Ego	2019
Cooking	CMU-MMAC	Cooking	-	5	186	186	39	RGB+D+Keypoints+Acc+Mic	Yes (RFID)	5	2010
	50 Salads	Cooking	4h	52	50	2,967	25	RGB+D+Acc	Yes (Acc)	1	2013
Procedural Activity	COIN	Instruction Video	476h38m	180	N/A	46,354	N/A	RGB (Youtube)	No	N/A	2019
	IKEA-ASM	Furniture Assembly	35h16m	33	371	16,764	48	RGB+D+Keypoints	No	3	2021
	Assembly101	Toy Assembly	42h+	202	362	1M+	53	RGB	No	12	2022
Industrial	InHARD	Industrial Actions	18h30m	14 + 72	38	4800	16	RGB+Keypoints (3D)	No	3	2020
	ABC Bento	Bento Packaging	3h22m	10	199	151	4	MOCAP	No	1	2021
	LARa	Picking	14h50m	8 + 19	324	8878	16	RGB+Keypoints (3D) +Acc+Mic	No	1	2020 +2022
		Packaging			125	2103	10				
	OpenPack	Packaging	53h50m	10 + 32	2048	20,129	16	D+Keypoints+LiDAR+Acc+Gyro +Ori+EDA+BVP+Temp	Yes	2	2022

∗When action classes are defined in a hierarchical manner, the numbers of classes in different levels are shown with separator "+".

3.2 Related Work

3.2.1 Datasets on Manual Tasks

Although many multimodal, vision-based, and IMU-based datasets for daily activity recognition have been made publicly available [84, 61, 13, 37], the number of publicly available datasets for work activity recognition in industrial domains remains limited. Table 3.1 summarizes the attributes of datasets on human activities and manual labor. To the best of our knowledge, the LARa dataset [59, 58] is the only dataset on work activity recognition in logistics. However, the activity classes used in the dataset do not reflect the complexity of work processes well, and the number of activity (operation) instances is limited, as mentioned in the introduction. In addition, the dataset does not provide readings from IoT-enabled devices or a rich set of metadata.

The InHARD dataset [16] and the ABC Bento packaging dataset [4] are designed to accelerate human-robot collaboration in industrial settings. The InHARD dataset consists of RGB and 3D keypoint data from 16 subjects collected while they were assembling various parts and components. The ABC Bento Packaging dataset is a dataset that captures activities related to bento packaging. This dataset focuses on common mistakes made by bento manufacturers, such as "forgetting to put in ingredients," and provides labels only for outlier types. The ABC Bento Packaging dataset is quite small to be applied to data-driven algorithms, such as deep learning. These vision-based datasets also lack sensor data modalities. In contrast, OpenPack is a large-scale dataset containing $20,129$ work operation instances with multimodal sensor data. The activity label set used in this dataset was designed based on operations specified in instruction documents used at actual logistic centers. In addition, OpenPack provides a rich set of metadata such as the size and item code (JAN code) of each item and a set of items to be packed.

Datasets of various complex procedural activities are also available. Specifically, many multimodal/vision-based cooking activity datasets are available such as CMU-MMAC [84], 50 Salads dataset [85], Breakfast Actions Dataset [38], EPIC-KITCHENS [17], and the Cooking Activity Dataset [39]. Vision-based datasets focused on procedural activities other than cooking include IKAE-ASM [10] and Assembly101 [79] for assembling furniture and toys, and COIN, a collection of instructional videos collected from YouTube. As noted above, many multimodal or vision-based datasets on manual tasks in daily life have been made publicly available. In contrast, the availability of public datasets for industrial domains remains limited, and OpenPack is the first large-scale dataset for activity recognition in industrial domains. This may be attributed to the difficulties in collecting datasets for industrial domains compared to everyday tasks. Collecting data for industrial applications requires close collaboration with industrial engineers working in an actual target environment to coordinate an experimental environment with various equipment for the target task, to define a set of activity labels by obtaining an actual work instruction document used in the target industrial environment, and to employ workers with experience in the target task as research subjects.

3.2.2 Sensor-Based Activity Recognition Methods

A number of activity recognition methods have been proposed. Methods that use IMU sensors include DeepConvLSTM [82], U-Net [109], and Conformer [32]. LOS-Net [103] and MGA-Net [52] were designed to recognize activities in industrial applications. Vision-based approaches have also been proposed, such as I3C [11], ST-GCN [97], and MVT [96]. Although vision-based methods are very powerful, packaging work is typically subject to occlusion caused by large items or boxes. Therefore, we evaluated the proposed method on IMU sensor data. Several sensor fusion methods have been proposed for multimodal activity recognition, such as Multi-GAT [30], and early/late fusion [54]. Data from IoT-devices are sparse but have a strong relationship with specific activity classes, which is quite different from other sensor data. However, these methods do not consider this property.

3.3 OpenPack Dataset

OpenPack[2] is the first multimodal large-scale dataset for activity recognition in industrial domains. 16 distinct participants packed 3,956 items in $2,048$ shipping boxes in total, and the total duration of our dataset is 53.8 hours, consisting of 104 sessions (see below). The total number of collected work operations and actions are $20,129$ and $52,529$ respectively. The average lengths of operations and actions are 9.2 and 4.1 seconds respectively. OpenPack is the largest industrial dataset that includes both vision and wearable sensor data with precise labels by annotators. The most similar dataset to OpenPack is LARa [59], but the total data duration of LARa is 14.8 h. Further details are introduced as follows.

3.3.1 Packaging Work

As shown in Fig. 3.1, a typical series of complex operations is iterated, with each iteration (i.e., period) comprising a sequence of operations such as assembling a shipping box and filling the box with items. In a given work period, a worker processes a single shipping order. That is, the worker picks items in a specified order, double-checks the items, assembles a shipping box, fills the box with the items, and so forth to complete the order. When performing specific operations, the worker uses IoT-enabled devices such as a handheld barcode scanner, and the operation is recorded and transmitted by the device. Because the size of items to be packed, the number of items, and the size of shipping items depend on shipping orders, sensor data collected in different work periods and the duration of the same operation in different work periods vary. The task of recognizing specific actions is challenging owing to these characteristics of packaging work.

[2]`https://open-pack.github.io/`

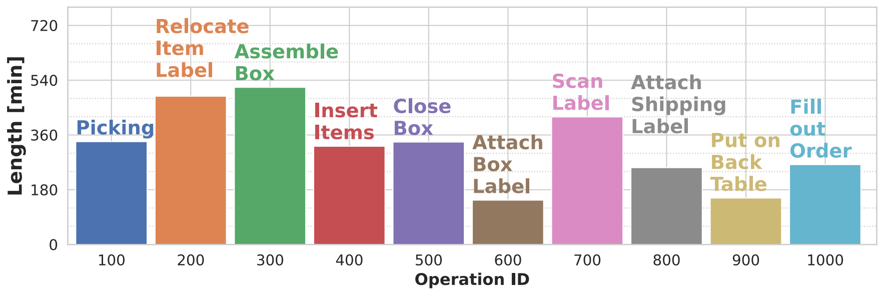


Figure 3.2: Total recording length of each operation

3.3.2 Data Construction

Classes

Operation classes used in OpenPack were defined based on an instruction document used in an actual logistics center. The document specifies a sequence of operations performed by a worker, and each worker in the center performs operations according to the document. Therefore, the basic activities performed by all workers, i.e., operations, were used to label the dataset. Our dataset contains ten classes of operations, including picking, relocating item labels, assembling boxes, inserting items, closing the box, attaching a label to the box, scanning the label, attaching a shipping label, placing the item on a rack, and filling out an order. Note that many other logistics centers also utilize patterns of operations very similar to those used in this study. Fig. 3.2 shows the distribution of the total recording length of each operation.

An instructional document also contains a description of each operation that explains how to perform the operation. For example, a description of the "relocate item label" operation is given as "*Remove the label from the items and place it on the bottom margin of the packaging list. Check the product name and quantity on the list and label with a ballpoint pen.*" Based on the description, we also defined a set of action classes included in each operation. For example, as shown in Fig. 3.1, the "assemble box" operation is composed of four actions, including "pick up cardboard," "bend flap," "attach tape," and "turn over box." Our dataset contains 32 action classes in total. The action classes are useful to enable a manager in a logistics center to assess the status of a job in progress in detail.

Subjects

We invited 16 subjects to participate in our data collection process. The ages of the subjects were ranged from 20 and 50. 12 subjects had experience in packaging work ranging from 1 month to 4 years. In addition, 4 subjects did not have work experience. One subject was left-handed. Each subject was assigned a consistent identifier throughout the entire dataset.

Sessions and Work Periods

In our dataset, a session is composed of iterations of work periods. In each work period, a subject processes the items on an order sheet. To complete the order, the subject performs the operations specified in the instruction document.

3.3.3 Data Collection

Collection Environment

We collected data in a dedicated environment shown in Fig. 3.3. With the help of industrial engineers, we constructed a 3m $\times$ 5m environment designed to simulate an actual workspace in a logistics center. The environment mainly comprised a workbench, a back table on which items were placed after being picked from shelves by another worker, boxes containing air cushions, and a trash can. The distance between the workbench and the back table was approximately 2 meters, and a worker worked between the bench and the back table. A handheld barcode scanner, a printer, and craft tapes used for packaging were located on the right side of the table. Four types of cardboard boxes were available for packing, including small, medium, large, and extra-large sizes, as shown in Fig. 3.3.

Data Modalities

OpenPack provides nine data modalities, including acceleration, gyroscope, quaternion, blood volume pulse (BVP), and electrodermal activity (EDA) data, temperature, as well as keypoints, a LiDAR point cloud, and depth images collected in the dedicated environment. Four IMU units were attached to the subject's left and right wrists and upper

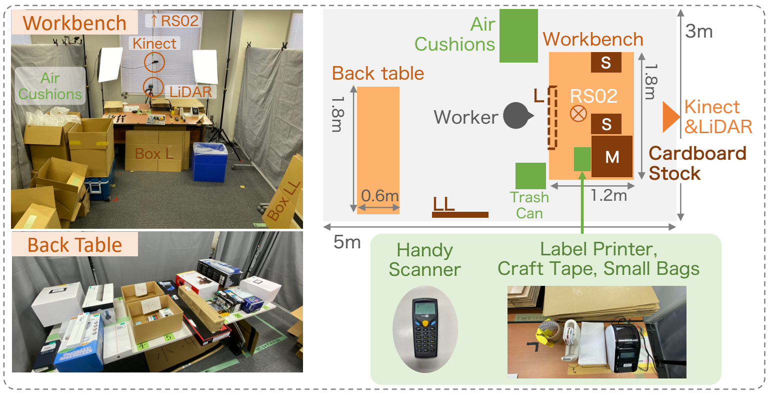


Figure 3.3: Environmental setup

arms to collect acceleration data on three axes, as well as gyroscope and quaternion data at 30 Hz. In addition, two Empatica E4 sensors were attached to the subject's left and right wrists to collect BVP and EDA signals at 64 Hz and 4 Hz, respectively, in addition to acceleration data at 32 Hz. Kinect and LiDAR sensors were installed as front-view cameras and RealSense (RS02) as a top-view camera as shown in Fig. 3.3. The LiDAR sensor was considered effective in accurately tracking the subject's position when they were away from the workbench.

OpenPack also provides operational logs of IoT-enabled devices, i.e., the handheld scanner and label printer, in the environment. The operational logs of the handheld scanner, for example, contain a time-stamp of a scan and an identifier of a scanned item. These are highly reliable sources of information for recognizing the scanning operation and may improve recognition performance.

Data Collection Procedure

Before data collection, each subject received instructions related to the outline of the experiment and the operations to be performed based on the instruction document. Subsequently, we obtained the informed consent of the subjects, who then practiced the packaging work by performing work periods up to five times. Subsequently, we activated and calibrated the sensors and attached the wearable sensors to the subjects.

The subjects iterated up to five data collection sessions within 6 h (including a 60-min lunch break). At the beginning of each session, the subject received 20 order sheets and then sequentially processed the sheets. That is, the subject completed 20 shipping boxes in the session. A 15-min break was included between two consecutive sessions. Note that each session was conducted under either one of the following four scenarios. After data collection, the data were labeled by annotators with 5-6 years of experience in human/animal activity data labeling by reference to the RGB images with the help of industrial engineers.

Scenarios

The difficulties in packaging work recognition depend on various factors in logistics centers. For example, (i) some experienced workers alter operational procedures to improve efficiency, (ii) the size and the number of items greatly affect workers' movements and sensor data, and (iii) workers sometimes perform irregular activities. To simulate these situations and factors, we prepared the following four scenarios. Scenario 1 was the most basic setup, in which workers followed instructions in principle. Scenario 2 allowed workers to make changes to the work procedures according to their own judgment and involved a richer variety of combinations by adding new items. Scenario 3 introduced irregular activities, such as picking items for several boxes at once. In Scenario 4, a sound alarm was introduced to hurry workers to simulate a busy situation.

Metadata

OpenPack provides a rich set of metadata, which is mainly composed of subject- and order-related metadata. The subject-related metadata contains information regarding each of the participants' experience in packaging tasks, as well as their dominant hand, gender, and age.

Order-related metadata contains information regarding an order sheet processed by a subject in a work period. OpenPack assumes an online order management system and provides information regarding an identifier of an order and a set of items to pack in the order. The management system also stores information regarding an identifier, product code, and the size of each item. These identifiers are used to manage items in an order management system. In contrast, product codes are unique numbers for each item, which are widely used in retail sales and enable information to be retrieved regarding an item, such as product name, product type, and price. OpenPack also provides this information.

In addition to the above order information, which can be automatically obtained from the order management system, OpenPack also provides manually annotated metadata regarding mistakes and accidents in each period. For example, operations that a subject forgot to perform and those that were conducted in the wrong order were recorded.

Potential Research Tasks

In addition to basic operation and action recognition using depth or wearable sensors, OpenPack enables various designs of research tasks, which includes the following: (i) transfer learning across workers/sensor positions, (ii) crossmodal transfer learning using, e.g., knowledge distillation [37], (iii) skill assessment using sensor data and metadata related to work experience as ground truth, (iv) failure detection using sensor data and metadata related to mistakes as ground truth, (v) counting the number of necessary actions or the number of packed items using sensor data, (vi) estimating workers' levels of fatigue using sensor and physiological data.

3.4 Ladder-Shaped Two-Stream Network with Sparse Anchoring (LTS-Net)

Based on the OpenPack dataset, we propose a new activity recognition model that effectively utilizes sparse readings from IoT-enabled devices. There is a strong connection between devices in use and a user's current activity. For example, when a subject uses a handheld scanner, the activity label for that moment would be "scan label." In this study, we leverage the high-confidence source, i.e., readings from IoT-enabled devices, as an *anchor* to precisely recognize activities (operations). To efficiently leverage anchors, we propose a LTS-Net for time-series classification (Fig. 3.4). The ladder-shape is composed of bidirectional feedback blocks (BF blocks) comprising two vertical streams, i.e., a feature stream and an anchor stream, and feedback branches that are used to exchange information between the streams. Sensor data (e.g., IMU data) are fed into the feature stream after initial feature extraction by the encoder, and feature

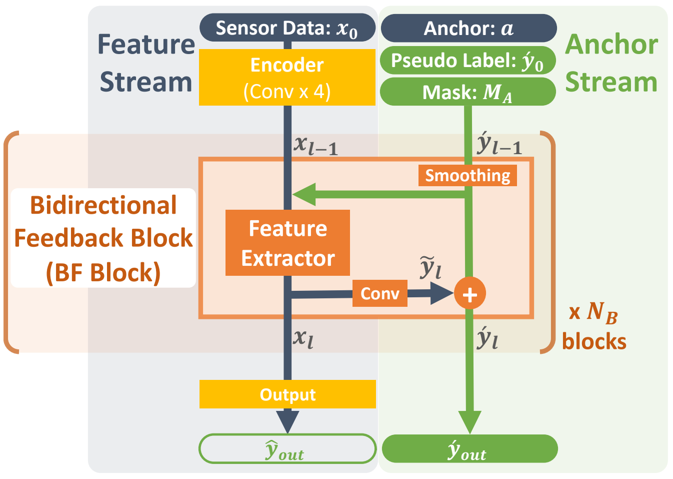


Figure 3.4: Architecture of Ladder-shaped Two-stream Network

extraction is performed. Anchors and pseudo-estimates of activity classes generated from the anchors are fed into the anchor stream, and the pseudo-estimates are refined in deeper layers. By performing feature extraction and activity class estimation in different streams, we can refine pseudo-estimates generated from high-confidence anchors by referring to sensor data while retaining the high-confidence information in deeper layers.

3.4.1 Preliminary

We assume that $\boldsymbol{x}_0$ is time-series sensor data, such as acceleration and skeleton data, and $\boldsymbol{a} \in \mathbb{R}^{N_A \times T}$ is derived from readings from IoT devices, i.e., anchors. The outputs of the model $\acute{\boldsymbol{y}}_{out} \in \mathbb{R}^{N_C \times T}$ are the probability of each activity (operation) label for each time step. N_A, N_C, and T denote the number of anchor channels, the number of activity classes, and the length of the time series, respectively. For example, when two types of usages of IoT-enabled devices are available, e.g., scanning with a handheld scanner and printing with a label printer, N_A is 2. When the handheld scanner (i-th channel) is used at time t, $a^{i,t}$ is set to 1. Otherwise, it is set to 0. In the example of the upper portion of Fig. 3.5, the use of the handheld scanner corresponds to the 1st channel and its corresponding value is 1 when it is used.

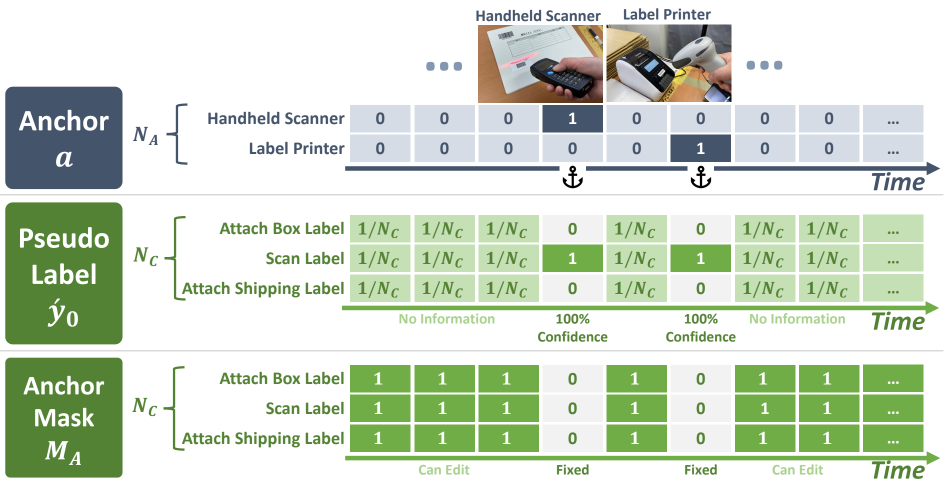


Figure 3.5: Examples of anchor, pseudo label, and anchor mask

3.4.2 Anchor Stream

To efficiently leverage anchors, an anchor sequence $\boldsymbol{a}$, a pseudo-label sequence $\acute{\boldsymbol{y}}_0$ generated from $\boldsymbol{a}$, and an anchor mask M_A are fed into the anchor stream; Fig. 3.5 shows an example. $\acute{\boldsymbol{y}}_0 \in \mathbb{R}^{N_C \times T}$ is an initial pseudo-estimate of class probabilities calculated from a set of possible activity classes at each time step t determined based on the anchor. We define a function that calculates the possible activity set at time t as $C(\boldsymbol{a}, t)$. For example, when the handheld scanner is used at time t, i.e., $\boldsymbol{a}^t = [1, 0]^{\mathrm{T}}$, $C(\boldsymbol{a}, t) = \{\mathrm{ScanLabel}\}$, which is determined based on labeled training data. In contrast, when no IoT devices are used at time t, i.e., $\boldsymbol{a}^t = [0, 0]^{\mathrm{T}}$, $C(\boldsymbol{a}, t)$ is a set of all the activity classes because we cannot determine specific possible activity classes based on $\boldsymbol{a}^t$. An initial estimate (pseudo-label) at time t for the c-th class is calculated as follows.

$$\acute{y}_0^{c,t} = \begin{cases} 1/|C(\boldsymbol{a}, t)|, & c \in C(\boldsymbol{a}, t) \\ 0, & \text{otherwise} \end{cases}$$

As shown in the middle panel of Fig. 3.5, when the handheld scanner was used, the element corresponding to *Scan Label* was 1, and the other elements were 0. When no IoT devices were used at time t, $|C(\boldsymbol{a}, t)|$ was identical to N_C and the class probability (pseudo-label) was uniform, i.e., $1/N_C$, among all the classes. $\acute{\boldsymbol{y}}_0$ is updated within the deeper layers in the anchor stream by referring to sensor data features.

Avoiding changing the high-confidence parts in the pseudo-labels is important in deeper layers. An anchor mask $M_A \in \mathbb{R}^{N_C \times T}$ is introduced to specify elements of $\acute{\boldsymbol{y}}_l$ to be updated in the BF block. An element of the mask for the c-th class at time t, i.e., $M_A^{c,t}$, is defined as follows.

$$M_A^{c,t} = \begin{cases} 0, & \acute{y}_0^{c,t} \in \{0, 1\} \\ 1, & \text{otherwise.} \end{cases}$$

As shown in the lower panel of Fig. 3.5, when probability 0 or 1 is assigned to the $\acute{\boldsymbol{y}}_0^{c,t}$, the corresponding element of the mask is set to 0 because we want to fix the high-confident estimate. Otherwise, we assign 1 to allow updating the low-confident estimate in the BF block.

3.4.3 Bidirectional Feedback Block (BF Block)

As shown in Fig. 3.4, LTS-Net comprises multiple BF blocks, each of which consists of a pair of feedback branches and a set of modules between them. BF blocks extract features from sensor data and update $\acute{\boldsymbol{y}}_0$ to make predictions by referring to the extracted features. The inputs to the l-th block are the outputs of the previous block: $\boldsymbol{x}_{l-1}$, $\acute{\boldsymbol{y}}_{l-1}$, and M_A. The output of this block is $\boldsymbol{x}_l$ and $\acute{\boldsymbol{y}}_{l-1}$, with the same shape as the corresponding input. $\acute{\boldsymbol{y}}_l$ from the last BF block is the final output of LTS-Net.

Smoothing is first applied to the input pseudo-labels $\acute{\boldsymbol{y}}_{l-1}$. This removes spontaneous activity transitions in the pseudo-labels and propagates the anchor information to the surroundings. The feature extractor is an arbitrary module that extracts features from sensor data. For example, convolutional layers, Conformer [32], a WPCP module [103], etc. can be used. We also use $\acute{\boldsymbol{y}}_{l-1}$ and $\boldsymbol{a}$ as additional inputs of the feature extractor to employ anchor information in the feature extraction. The output of the feature extractor $\boldsymbol{x}_l$ is transformed to the same shape as $\acute{\boldsymbol{y}}_{l-1}$ by a 1D convolution and sigmoid function to output $\tilde{\boldsymbol{y}}_l \in \mathbb{R}^{N_C \times T}$, which is an estimate based mainly on sensor data. Then, $\acute{\boldsymbol{y}}_{l-1}$ is updated by using $\tilde{\boldsymbol{y}}_l$ as follows.

$$\acute{\boldsymbol{y}}_l = (1 - \beta M_A)\acute{\boldsymbol{y}}_{l-1} + \beta M_A \tilde{\boldsymbol{y}}_l.$$

With this operation, the pseudo-label sequence is modified in each block while preserving anchor information. β controls the amount of update within a single block.

Table 3.2: F1-measures of methods

F1-measure	Macro Avg.				Weighted Avg.
User	U0104	U0108	U0110	All	All
DCL-SA	0.762	0.782	0.643	0.752	0.766
DCL-SA (Early)	0.810	0.815	0.630	0.760	0.780
DCL-SA (Late)	0.820	0.807	0.725	0.797	0.813
Conformer	0.801	0.724	0.676	0.748	0.752
Conformer (Early)	0.834	0.778	0.716	0.787	0.797
LOS-Net	0.790	0.793	0.611	0.755	0.762
LOS-Net (Early)	0.819	0.815	0.675	0.784	0.802
LOS-Net (Late)	0.816	0.813	0.644	0.777	0.787
LTS-Net (WPCP)	**0.847**	**0.842**	**0.757**	**0.830**	**0.846**

3.4.4 Training

The loss function of LTS-Net is calculated based on the softmax cross-entropy $\mathrm{CE}(\cdot)$ computed on the output of the anchor stream $\acute{\boldsymbol{y}}_{out}$ and the feature stream $\hat{\boldsymbol{y}}_{out}$ as follows:

$$\mathcal{L} = \mathrm{CE}(\boldsymbol{gt}, \acute{\boldsymbol{y}}_{out}) + \gamma \mathrm{CE}(\boldsymbol{gt}, \hat{\boldsymbol{y}}_{out}),$$

where $\boldsymbol{gt}$ denotes the ground truth and γ is a hyperparameter that controls the impact of $\hat{\boldsymbol{y}}_{out}$. Adam optimizer with the cosine annealing learning rate scheduler [43] is used for training.

3.5 Evaluation

3.5.1 Evaluation Methodology

We evaluated LTS-Net on the OpenPack dataset. The data collected in Scenario 1 were used for evaluation, and operation recognition (semantic segmentation) was performed. Acceleration data from an IMU attached to the left wrist were used as the sensor data. Subjects U0104, U0108, and U0110 were used as test subjects and the remaining subjects were used as training subjects. The performance of our model was evaluated in terms of F1-measure. Because the ground truth was provided at 1 Hz, the outputs of the model were downsampled to 1 Hz before computing the metrics. We compared our

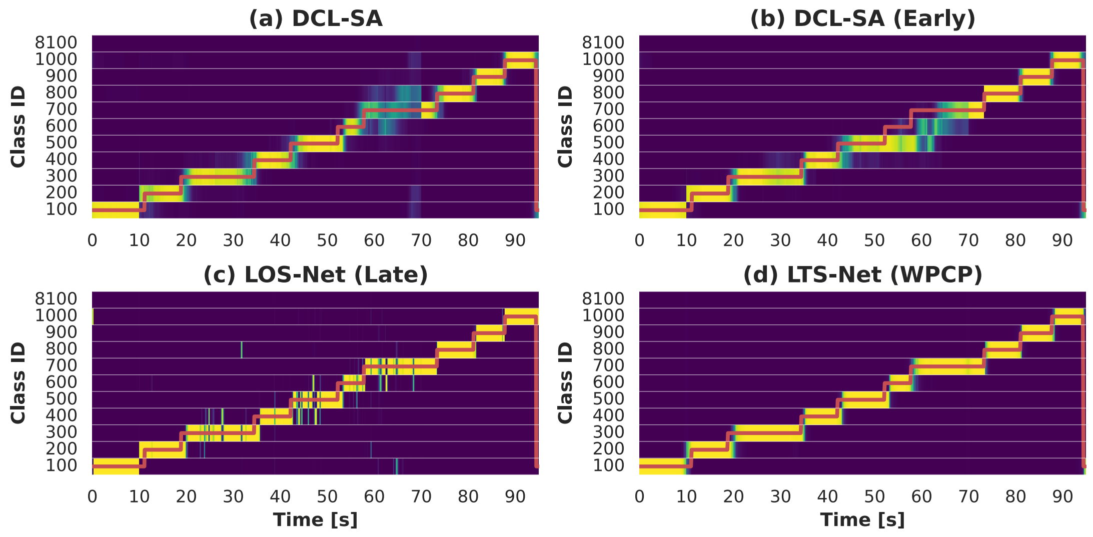


Figure 3.6: Examples of time-series data of ground truth (red line) and class estimates, i.e., probabilities. High- and low-class probabilities are yellow and blue, respectively.

LTS-Net to DeepConvLSTM + Self-Attention (DCL-SA) [82], Conformer [32], and LOS-Net [103], which use only acceleration data as inputs. We also prepared early/late fusion models for each method, which also use IoT data. As a feature extractor of the LTS-Net, we used WPCP module and the number of BF block was set to 8. The model parameters described in the original papers were used directly, whereas the optimization parameters, such as the learning rate, were tuned.

3.5.2 Results

Table 3.2 shows the recognition performance on the test set of the models trained by changing the random seed 5 times. The F1-measure (macro avg) is approximately 0.75 for all the existing methods when data from IoT-enabled devices are unavailable. By adding data from IoT-enabled devices, the F1-measures of DCL-SA, Conformer, and LOS-Net improved by up to 0.04. The F1-measre of DCL-SA (Late), which is the best score among the existing methods, was 0.797 when data from IoT-enabled devices were used. The proposed LTS-Net (WPCP) method, which employs the WPCP module as the feature extractor, significantly outperformed DCL-SA (Late) in terms of F1-measure by 0.033 ($p = 0.019$ by Welch's t-test) and achieved 0.830 by effectively leveraging anchors. In particular, only LTS-Net achieved a score higher than 0.75 for U0110, whose skill level was Expert and work speed is much higher than the other training subjects. The same trend can be observed for the F1-measure (weighted average).

Figure 3.6 shows examples of model estimates. DCL-SA, which does not use anchors, failed to recognize the latter half of the scan label operation (Class ID: 700). Although DCL-SA (Ealry) and LOS-Net (Late) employed anchors related to a hand scanner, the estimates of the scan-label operation and surrounding operations were unstable compared with those of LTS-Net, indicating the ability of LTS-Net to efficiently fuse IoT and sensor data.

LTS-Net outperformed LOS-Net proposed in Chapter 2. LOS-Net assumes a more realistic environment with limited training data. In contrast, LTS-Net was developed for an envionment with a large training dataset. Therefore, LTS-Net outperformed LOS-Net on the OpenPack dataset. However, LTS-Net employs the LOS-Net's WPCP module as a feature extractor, which was proposed to recognize work processes that consist of a sequence of actions. This LTS-Net result again showed that the WPCP module is effective in recognizing work processes.

3.6 Conclusion

This study presented a new large-scale dataset for packaging work recognition called OpenPack. OpenPack contains 53.8 hours of multimodal sensor data, including depth images, IMU data, and readings from IoT-enabled devices. This dataset enables researchers to explore more challenging activity recognition methods for industries, such as collaboration with IoT devices. Furthermore, we propose LTS-Net to effectively utilize sparse readings from IoT devices. The evaluation results on our dataset have demonstrated the effectiveness of the proposed two-stream architecture.

Chapter 4

Toward Understanding Acceleration-based Activity Recognition Neural Networks with Activation Maximization

In this chapter, we describe a supporting method for understanding the algorithm of human activity recognition neural network. We focus on a visualization method called Activation Maximization and explore the method to apply this technology for the human activity recognition model. We also propose one possible solution to overview the concepts learned by the neural network.

4.1 Introduction

Owing to the recent proliferation of wearable devices, activity and gesture recognition techniques using inertial sensors in the devices have been actively studied. These techniques have many potential smart applications such as supporting elderly care, healthcare, and home automation. Although recent studies on activity and gesture recognition have relied on deep learning [63, 54], the recognition process in neural networks has been regarded as a black box. In the field of sensor-based activity recognition, some studies have attempted to tackle this problem and visualize the inside of activity recognition neural networks. Zeng et al. and Ma et al. visualized a model with an attention mechanism by drawing an attention map [107, 45]. Saeed et al. highlighted regions in the input time-series that contributed to the model's outputs using gradient information. Although these methods can indicate important regions focused by the model, they cannot reveal what features the models are extracting.

We believe that the techniques for visualizing the features learned by activity recognition neural networks significantly contribute to researchers and developers of human activity recognition methods and models as follows.

- **Activation Atlas for human activity**: An activation atlas [12] is an explorable map of visual features that a network for vision-based object recognition uses to recognize objects. In the case of activity recognition networks, we can generate a map of acceleration data segments that show a universe of human activity concepts, representing inherent relationships between activities and atomic human motions constituting complex human activities. It can also contribute to explainable activity/gesture recognition, e.g., gesture-based authentication.

- **Model validation**: By visualizing the acceleration features extracted by a unit in a convolutional neural network (CNN), we can easily determine if the network is properly trained. For example, when the generated signals from many different units in the CNN are similar, we can deduce that the network is redundant and that a smaller network is suitable for the training data. We believe that such information is useful for fine-tuning the CNN parameters such as the number of units and network depth.

In the computer vision (CV) research community, techniques related to understanding the inside of a neural network have been extensively studied to interpret network behavior [40, 105, 48, 57, 76, 55, 81]. One of the promising techniques for analyzing neural networks is activation maximization (AM) [19], which is used to find an input image that maximizes the activation of a unit in a neural network to reveal the concept learned by the unit [40, 105]. However, this method can generate images that exhibit extreme activations but are unrecognizable by the human eye [56, 51]. Fig. 4.1 (1) shows an acceleration signal generated by a naive AM using an activity recognition network and the corresponding activation values. The dotted line indicates the 95th percentile

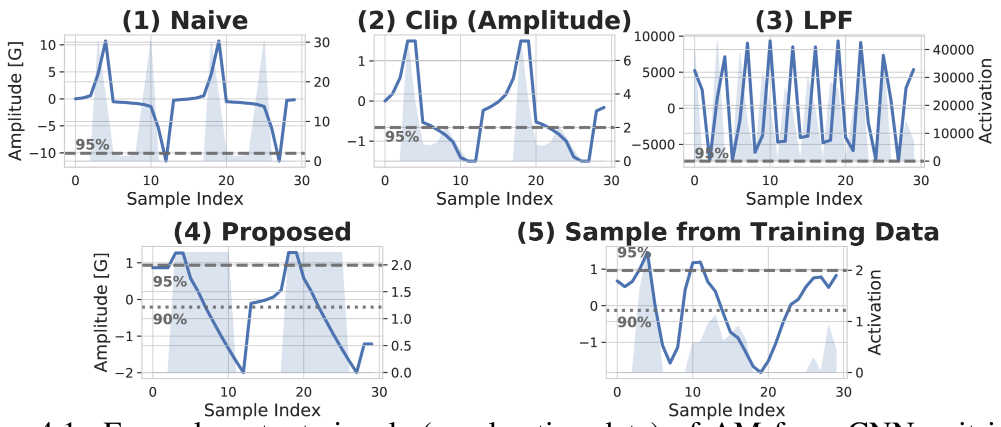


Figure 4.1: Example output signals (acceleration data) of AM for a CNN unit in an activity recognition network using (1) naive AM, (2) a method with amplitude clipping, (3) the method proposed by [101], and (4) the proposed method. Fig. (5) shows an actual signal found in the training dataset. The horizontal axis shows the sample index. The background histogram and right vertical axes show the activation values of the corresponding samples.

of the distribution of activation values for the corresponding unit when feeding all the training data. The signals generated by the naive AM exhibit activation values that are about 15 times larger than those of the training data, resulting in unrealistic extreme amplitudes (10G) in the segments corresponding to the extreme activation values.

In the CV studies, various techniques for addressing this issue have been investigated (e.g., regularization techniques [40, 105], introducing learned priors [55]). However, these techniques have been developed for image-based object recognition neural networks and have not yet been applied to acceleration-based activity recognition. For example, because the brightness value range of an RGB pixel of an image is restricted, i.e., between 0 and 255, we can perform optimization constrained to the value range, e.g., using clipping, to suppress meaningless noises. However, it is difficult to restrict the value range of generated signals using a certain threshold value. This is because, first of all, actual acceleration data has a smaller value range than the physical limit such as the measurement range. Therefore, using the physical limit as a threshold value may result in generating signals with extreme values that are not observed in normal human movements. In addition, since each unit has different sensitivities to the amplitude of the input signal, it is difficult to cover all units with a single threshold value. Fig. 4.1 (2) shows a signal generated using a clipping technique where the data points in generated signals whose amplitudes exceed a threshold ($= \pm 1.5$G; approximately 95% point of acceleration values in the training data) are ignored, indicating the difficulties in suppressing extreme activation values. Therefore, techniques to suppress meaningless noise with large amplitudes in acceleration data are required.

Therefore, this study investigates AM optimization techniques that are suitable for acceleration-based activity recognition. We propose novel regularization techniques that suppress extreme amplitudes by restricting the activation values in a feature map. The primary contributions of this study are as follows. (i) To the best of our knowledge, this is the first work that leverages activation values to regularize AM for neural networks. (ii) We propose new AM techniques for human activity recognition networks: 1) a regularization term that penalizes extreme activations generated in the AM process, and 2) activation clipping that suppresses the generation of extreme activation. By leveraging the proposed techniques, we can generate natural signals with reasonable activation values, as shown in Fig. 4.1 (4). (iii) We evaluated the proposed method qualitatively and quantitatively using publicly available datasets, and it was found that the proposed method could generate signals that are similar to those that actually appeared in a training dataset (see Fig. 4.1 (4) and (5)).

4.2 Related Work

Investigating the decision criteria of a neural network is important for validating and improving the network. Previous researches in the field of wearable sensor-based activity recognition have attempted to visualize action recognition neural networks in order to reveal the decision criteria of the network, which is considered a black box. Zeng et al. and Ma et al. proposed models with attention mechanisms to focus on important modalities (e.g., important body parts where IMUs are attached) [107, 45]. They visualized the attention map to validate the behavior of the attention layer. Saeed et al. [73] introduced self-supervised learning to reduce the amount of labeled data required for network training. They used a saliency map [81] to compare a self-supervised network and a fully supervised network. A saliency map can visualize regions of the input signal (i.e., time segments), which greatly contributes to the network's prediction using gradient information. Fukui [22] also proposed a similar visualization technique that does not use gradient information. Although a unit is the most fundamental element that performs feature extraction, these methods can only locate the region of the input signal (e.g., time segments and body parts) that the network has focused on, and cannot explain the extracted concept at the unit level.

Visualization techniques for neural networks have also been actively studied in the CV research field [81, 78, 48]. Among the various visualization methods, we focus on activation maximization (AM) [19] in this study to obtain unit-level visualization. AM is a technique to find an input that maximizes the activation of a unit in the network by employing gradient ascent (see Section 4.3.2 for details). AM has been used to visualize concepts related to an output class [105, 55] and the function of an intermediate unit [40, 105]. In this study, we focus on time-series acceleration data and attempt to investigate the function of each unit in the trained model by employing AM.

AM tends to produce signals with high-frequency noises. Yoshimura et al. [101] performed a preliminary investigation on using AM for activity recognition networks. They simply employed a low-pass filter to reduce high-frequency noise. However, in many cases, signals generated by their method had unnaturally large amplitudes as shown in Fig. 4.1 (3). This might be because applying LPF in each iteration significantly alters the waveform, which is not suitable for gradient methods like AM that gradually update the waveform. In contrast, we introduce/propose regularization techniques for AM to generate natural signals.

Some prior studies generated acceleration data based on a generative adversarial network [90, 83], which are used as additional training data. However, these methods just generated signals similar to actual training data; they did not generate sensor data features that a neural network learns.

4.3 Synthesizing Acceleration Data

4.3.1 Preliminaries

In this study, we assume three-axis acceleration sensors attached to a user. We also assume a CNN-based neural network to recognize the user's daily activities. Note that this neural network is trained in advance on a daily activity dataset. The input to the neural network $\boldsymbol{X}$ is a sensor data segment within a sliding time window of length N_T. The intermediate output of unit u in the l-th layer is denoted as $\boldsymbol{X}^{(l,u)}$. $\boldsymbol{X}$ is an $N_S \times N_T$ matrix, and $\boldsymbol{X}^{(l,u)}$ is an $N'_S \times N_T$ matrix. Note that N_S is the total number of input sensors (axes), and N'_S is a constant that depends on the size of the convolutional filters.

4.3.2 Activation Maximization (AM)

As mentioned above, AM is a technique to find an input that maximizes the activation of a network unit by employing gradient ascent, thereby making it possible to reveal the feature maps extracted by the unit. We obtain an input $\boldsymbol{X}^*$ that maximizes the activation of a unit u in a network as follows:

$$\boldsymbol{X}^* = \underset{\boldsymbol{X}}{\arg\max}\, f^{(l,u)}(\boldsymbol{X}) \tag{4.1}$$

Note that $f^{(l,u)}(\boldsymbol{X})$ is a function that calculates a representative value of the feature map of unit u in the l-th layer $\boldsymbol{X}^{(l,u)} \in \mathbb{R}^{N'_S \times N_T}$.

However, AM iteratively updates $\boldsymbol{X}$ to maximize the activation, starting from a randomly generated $\boldsymbol{X}$, which makes generating interpretable signals difficult. Specifically, AM suffers from the following two problems. (i) The generated $\boldsymbol{X}^*$ is reported to contain high-frequency noise [105][1]. (ii) It is possible to generate inputs $\boldsymbol{X}^*$ that are meaningless/unrecognizable to humans but exhibit high activation values [56].

To address these issues, Le et al. [40] reduced the noise when solving Equation 4.1 by introducing a constraint, with the norm of $\boldsymbol{X}$ being 1 ($||\boldsymbol{X}||_2 = 1$). Yosinski et al. [105] reduced the high-frequency components by introducing a regularization term into Equation 4.1 as follows:

$$\boldsymbol{X}^* = \underset{\boldsymbol{X}}{\arg\max} \left(f^{(l,u)}(\boldsymbol{X}) - wR(\boldsymbol{X}) \right) \tag{4.2}$$

where $R(\boldsymbol{X})$ is the regularization term. They used the Lp norm and total variation [74] as the regularization term. The Lp norm $R_{Lp}(\boldsymbol{X})$ and total variation $R_{TV}(\boldsymbol{X})$ are

[1]To the best of our knowledge, the reason for such high-frequency noises has not yet been investigated thoroughly.

calculated as follows:

$$R_{Lp}(\boldsymbol{X}) = \left(\sum_{s}^{N_S-1} \sum_{t}^{N_T-1} |x_{s,t}|^p \right)^{1/p}, \tag{4.3}$$

$$R_{TV}(\boldsymbol{X}) = \left(\sum_{s}^{N_S-1} \sum_{t}^{N_T-2} |x_{s,t} - x_{s,t+1}| \right), \tag{4.4}$$

where $x_{s,t} \in \boldsymbol{X}$ represents the sensor reading at sensor s and time t. In this study, we propose a regularization term suitable for acceleration data and modify the objective of AM (Equation 4.2).

Equation 4.2 is solved by employing the gradient method in a manner similar to neural network training. To train a neural network, gradient descent is employed to minimize the value of the loss function. In contrast, AM employs gradient ascent to maximize the value of $f^{(l,u)}(\boldsymbol{X}) - wR(\boldsymbol{X})$. First, we randomly initialize $\boldsymbol{X}$. Then, $\boldsymbol{X}$ is updated in gradient ascent as follows:

$$\boldsymbol{X} \leftarrow \boldsymbol{X} + \eta \frac{\partial}{\partial \boldsymbol{X}} \left(f^{(l,u)}(\boldsymbol{X}) - wR(\boldsymbol{X}) \right) \tag{4.5}$$

where η is the learning rate. We obtain the final result $\boldsymbol{X}^*$ by iterating the updating process until convergence.

4.3.3 Proposed Regularization Techniques to Suppress Extreme Activations

AM can generate unrecognizable signals; however, it exhibits extreme activations. For example, let us assume a unit in the first CNN layer with all element values in its filter being positive. Here, the ReLU function is used as the activation function. In this case, the amplitude of the input signals $\boldsymbol{X}$ is proportional to the activation value of the unit; thus, AM can generate signals with a significantly large amplitude. However, as mentioned above, it is difficult to restrict the value range of the acceleration data to be generated (i.e., clipping). In addition, because different units in the network have different sensitivities in terms of input signal amplitudes, we cannot define a value range for clipping based on the unobservable sensitivity of each unit. Therefore, we propose to leverage activation values in a feature map (i.e., $\boldsymbol{X}^{(l,u)}$) to avoid generating such extreme signals. Because we can know the distribution of activation values for each unit when we feed all the training data, we can easily define an adequate activation value range for the unit based on the distribution. When an extreme activation value is observed in a feature map in the iterations of the AM process, we can regard that a segment in the input signals surrounding the extreme activation value is abnormal and differs considerably from the signals observed in the network's training data.

Therefore, this study specifies a threshold for the activation values ($th_{eap}^{(l,u)}$) for each unit that is used to define extreme activation values. For example, in this study, this threshold is determined based on the two-sigma rule using the activation values of the unit when feeding all the training data[2]. Under the assumption that activation values are normally distributed, this threshold roughly corresponds to the 95th percentile of the activation values in the training data.

We propose a novel regularization term called "extreme activation penalty" (**EAP**) that leverages the activation threshold, which is described as follows.

$$\begin{aligned} &R_{EAP}(\boldsymbol{X}^{(l,u)}) \\ &= \mathrm{Mean}\Big(\Big\{a_{s,t}^{(l,u)} - th_{eap}^{(l,u)} | a_{s,t}^{(l,u)} \in \boldsymbol{X}^{(l,u)}; a_{s,t}^{(l,u)} > th_{eap}^{(l,u)}\Big\}\Big) \end{aligned} \tag{4.6}$$

where $a_{s,t}^{(l,u)} \in \boldsymbol{X}^{(l,u)}$ represents the activation value at sensor s and time t. When some activation values exceed the threshold, i.e., two-sigma of the activation values of the training data, the averaged exceeded value is used as a penalty. As above, EAP directly leverages the extreme activation values to suppress abnormal segments.

In addition, we introduce a second technique called "activation clipping" to suppress extreme activation values that are difficult to handle by EAP. In the normal AM (Equation 4.1), a representative value, i.e., the mean, is calculated by $f^{(l,u)}$; therefore, it is difficult to suppress an individual extreme activation value in $\boldsymbol{X}^{(l,u)}$ even when EAP is used. To address this issue, we apply the following clipping function to each value in the feature map before calculating the mean of the activation values.

$$clipping(a_{s,t}^{(l,u)}) = \begin{cases} th_{eap}^{(l,u)} & (a_{s,t}^{(l,u)} > th_{eap}^{(l,u)}), \\ a_{s,t}^{(l,u)} & (otherwise). \end{cases} \tag{4.7}$$

By using the above activation clipping, activation values larger than the threshold are

[2]The mean and standard deviation are calculated over non-zero activation values.

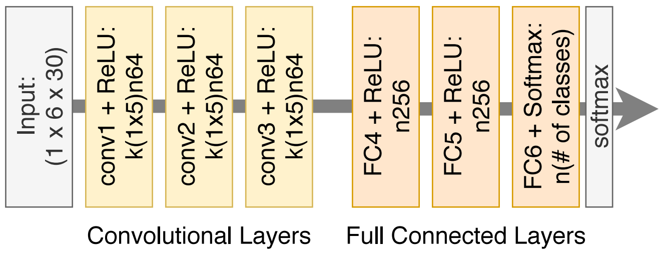


Figure 4.2: Network architectures used in the experiment: "$k(5 \times 1)$" and "$n64$" represent the convolutional filter size (along with the time axis) and the number of units in the layer of interest, respectively.

ignored, thereby preventing these activation values from becoming larger in the subsequent iterations of the AM process.

Our proposed method employs EAP in addition to the Lp norm and total variation in $R(\boldsymbol{X})$, and employs activation clipping.

4.4 Evaluation

4.4.1 Dataset and Activity Recognition Model

We employed two publicly available datasets to evaluate the proposed method: Daily and Sports Activities Dataset (**DSADS**) [9] and mHEALTH dataset (**MHEALTH**) [8].

DSADS

Barshan et al. [9] collected human activity data (19 activity classes), e.g., data for daily activities and sports, from eight participants using IMUs attached to the chest, arms, and legs. In this experiment, the data from participants 7-8 were used as test data, the data from participant 6 were used as validation data, and the data from the remaining participants were used as training data. We used three-axis acceleration data from sensors attached to the right hand and left leg, i.e., six sensors in total.

MHEALTH

Banos et al. [8] collected human activity data (12 activity classes), e.g., data for simple daily activities and exercise, from 10 participants using IMUs and ECG. In this experiment, the data from participants 9-10 were used as test data, the data from participant 8 were used as validation data, and the data from the remaining participants were used as training data. We used three-axis acceleration data from sensors attached to the right lower arm and left ankle, i.e., six sensors in total.

4.4.2 Activity Recognition Model

For preprocessing, we down-sampled the data to 30 Hz and applied min-max normalization. We then applied sliding time windows with a window size of 30 samples ($N_T = 30\mathrm{pt} = 1\mathrm{sec}$) and 50% overlap, which were fed into a neural network.

Figure 4.2 shows the network architecture used in this study. We designed our network based on the shared-filter hybrid fusion (SF-HF) model [54]. The models were trained using Adam optimizer [33] for 200 epochs with an initial learning rate of 10^{-4}. The models were trained five times with different random seeds. The f1-measures (and standard deviation σ) for the two datasets were 0.737 ($\sigma = 0.005$) and 0.955 ($\sigma = 0.004$), respectively. These scores are the average of the five runs. Based on the results of neural network-based activity recognition [91, 63], the networks appeared to be properly trained.

4.4.3 Procedures of Activation Maximization

AM was performed as described in the Synthesizing Acceleration Data section (Section 4.3.2). We focus only on a single sensor when running AM because the SF-HF model extracts features from each sensor independently. A 2-Hz sine wave (amplitude of 0.3 G) was used as the initial $\boldsymbol{X}$. The update step, i.e., Equation 4.5, was repeated 5,000 times with an initial learning rate η of 10^{-2}. The learning rate was decayed by multiplying it with $\gamma = 0.999$ after every iteration. The weight of $R(\cdot)$ was set to $w = 0.1$. These parameters were empirically selected.

4.4.4 Methods

To investigate the effectiveness of our AM techniques, we prepared a regularization term that penalizes according to the amplitudes of the signals to be generated (**AAP**). AAP is short for "Abnormal Amplitude Penalty," and this term directly restricts the value range of the acceleration data to be generated by using a threshold of the amplitude of acceleration data (th_{aap}). For simplicity, we used $th_{aap} = 1.5$ G for all datasets. (The upper limit of the two-sigma rule for **DSADS** and **MHEALTH** are 1.37 G and 1.57 G, respectively.)

For evaluation, we prepared the following methods. (1) LPF is applied according to [101] (**LPF**). (2) Only the L2 norm and total variation are used (**L2+TV**). This is the baseline method. (3) L2 norm, total variation, and AAP are used (**L2+TV+AAP**). (4) L2 norm, total variation, and EAP are used (**L2+TV+EAP**). (5–6) Applying activation clipping to methods (2) and (4), respectively (**L2+TV+clip** and **L2+TV+EAP+clip**). L2+TV+EAP+clip is our proposed method.

4.4.5 Evaluation Methodology

Previous studies on the visualization of neural networks did not perform quantitative evaluations [40, 101]. Instead, they conducted qualitative evaluations and demonstrated the effectiveness of the proposed method through application examples. We qualitatively evaluated our proposed method by following the previous studies.

In addition, we performed quantitative evaluation. To quantitatively evaluate the signals generated by the AM method, we determined whether the proposed method generates signals that actually appear in the training data. The signals generated by AM directly represent acceleration signals that a unit of interest in a CNN layer attempts to detect in the training data. Therefore, if we can find a real waveform similar to the signal generated by the AM, we can say that the method was able to generate a natural waveform that was actually observed in the training data. In contrast, if we are unable to find signals in the training data that are similar to the generated signals, we consider that the method demonstrates poor performance. Thus, by comparing the generated signals with actual signals in the training data, we quantitatively evaluated the quality of signals generated by the method.

We calculate the distance $d(u, i)$ between the generated signals for unit u and the segment extracted from the training data ($\boldsymbol{X}_i$) using the mean squared error (MSE) with sliding windows. Then, we select the top-N_{sim} similar segments based on the distance and calculate the average among the selected segments, which is used as the evaluation metric (S_{sim}). If an AM method generates signals that are similar to actual training signals, S_{sim} will be small.

The distance $d(u, i)$ between the generated signals for unit u ($\boldsymbol{X}^*$) and a training segment ($\boldsymbol{X}_i$) is defined as follows:

1. We compute the total activation value in each sliding time window of length $s_w (< N_T)$ using a feature map $\boldsymbol{X}^{(l,u)}$ of $\boldsymbol{X}^*$. The window size s_w is defined as the receptive fields of the convolutional layer of interest [6]. [3]
2. We select the time window (length is s_w) with the largest total activation value and extract the corresponding part of the signals from $\boldsymbol{X}^*$. This segment is denoted as $\hat{\boldsymbol{x}}^*$.
3. We also extract segments of length s_w from $\boldsymbol{X}_i$ using sliding time windows. We denote the segment extracted from time j to time $(j + s_w - 1)$ as $\hat{\boldsymbol{x}}_{i,j}$.
4. We calculate the MSE between $\hat{\boldsymbol{x}}^*$ and each segment from $\boldsymbol{X}_i$ (i.e., $\hat{\boldsymbol{x}}_{i,j}$).
5. The smallest MSE is defined as the distance ($d(u, i)$) between $\boldsymbol{X}^*$ and $\boldsymbol{X}_i$.

The distance $d(u, i)$ is described as follows.

$$d(u, i) = \min_j \{MSE(\hat{\boldsymbol{x}}^*, \hat{\boldsymbol{x}}_{i,j})\}, \tag{4.8}$$

In many cases, the length of the characteristic waveform is shorter than the window size of the generated signal. Therefore, we should focus on the important part of the generated signal, i.e., $\hat{\boldsymbol{x}}^*$. A segment with higher activation values is cropped from the generated signals based on the values of the corresponding activation in Step.1 and Step.2.

Finally, the evaluation metric S_{sim} is calculated as follows.

$$S_{sim} = \frac{1}{U} \sum_{u=0}^{U-1} \left[\frac{1}{N_{sim}} \sum_{i=0}^{N_{sim}-1} d(u, i) \right], \tag{4.9}$$

[3] $w_s = 5, 9, 13$ for 1st, 2nd, and 3rd layer respectively

where U represents the number of units in the layer of interest in which the amplitude of the generated signals is smaller than th_{fail}.

Some naive AM methods tend to fail to optimize signals, which results in generated signals with considerably large amplitudes. When we compute the evaluation metric S_{sim} for each layer, we ignore the MSE of the failed signals because these huge errors obscure the errors of successfully generated signals. We consider generated signals with amplitudes greater than th_{fail}, which is defined as the maximum amplitude in the training data, as a failure.

In this experiment, th_{fail} was set to 10 G based on the largest amplitude in the two datasets (i.e., 9.57 G). For N_{sim}, we set values corresponding to 1% of the total training samples.

4.5 Results

To demonstrate the effectiveness of the EAP and activation clipping, we conducted quantitative and qualitative evaluations.

4.5.1 Quantitative Analysis

Table 4.1 shows the results of the quantitative evaluation. S_{sim} (MSEs) is the average over five training sessions with different random seeds. "Conv1" shows the score for the first convolutional layer. The numbers in parentheses indicate the average number of failed units within 64 units over five trials. As shown in the table, introducing EAP and activation clipping significantly reduced the MSE. In particular, the MSEs and the number of failed units for the proposed method are much smaller than those for L2+TV and achieved the best scores, indicating that the proposed method generated signals similar to those actually found in the training data. We calculated the score S_{sim} with different N_{sim} (0.1%, 5%, 10%) and got the similar scores. This indicates that N_{sim}, a hyper parameter of the evaluation metrics, has a small effect on the score.

EAP and activation clipping were introduced to control the activation values. In order to confirm the effectiveness of them, we calculated the ratio of the maximum activation within generated signals from each unit to the threshold ($th_{eap}^{(l,u)}$). Fig. 4.3 shows the results, with each bar showing the mean value over all units in each layer. In

Table 4.1: Results of quantitative evaluation. Values in the table are S_{sim}, similarity measure between the generated signals and the training singals that is defined in Section 4.4.5. Smaller value is better. top-N_{sim} is set to 1%.

Dataset	Method	Conv1	Conv2	Conv3
DSADS	LPF	NaN (64.0)	3.242 (35.2)	1.534 (36.8)
	L2+TV	9.960 (2.4)	4.882 (1.6)	1.441 (24.6)
	L2+TV+AAP	8.060 (1.8)	3.841 (1.4)	1.616 (22.8)
	L2+TV+EAP	6.643 (0.6)	3.213 (1.4)	2.005 (22.2)
	L2+TV+clip	0.519	0.240	0.379
	L2+TV+EAP+clip	0.485	0.213	0.321
MHEALTH	LPF	NaN (64.0)	3.356 (36.0)	2.249 (37.0)
	L2+TV	7.841	7.919 (1.8)	2.292 (11.2)
	L2+TV+AAP	6.438	7.201 (1.4)	2.672 (10.2)
	L2+TV+EAP	5.874	7.287 (0.4)	3.283 (9.4)
	L2+TV+clip	1.044	0.182	0.226
	L2+TV+EAP+clip	1.021	0.157	0.197

many cases, the ratio values of the proposed method are closest to 1.0, indicating that the proposed method seldom generates signals with extreme activation larger than the threshold. This result also indicates that the proposed method is successful in controlling the activation values. These results show the effectiveness of using feature map activation values in the regularization.

Figure 4.4 shows examples of signals generated by the proposed method and their most similar training segments, as well as the time series of the activation values when the signals are fed into the unit. The overlaid blue segments are segments with high activation values, showing characteristic segments learned by the unit. Fig. 4.4 suggests that the proposed method could extract characteristic patterns that actually appear in human activity (e.g., an downward slope in “Stairs Up” and a horn-like shape in “Cycling”).

4.5.2 Qualitative Analysis

Figure 4.5 right shows examples of signals generated by L2+TV+AAP and the proposed method. This unit appears to extract a decrease in the acceleration data. Note that the activation values for L2+TV+AAP at index $= 5, 9, 20, 26$ are very large. The dotted line

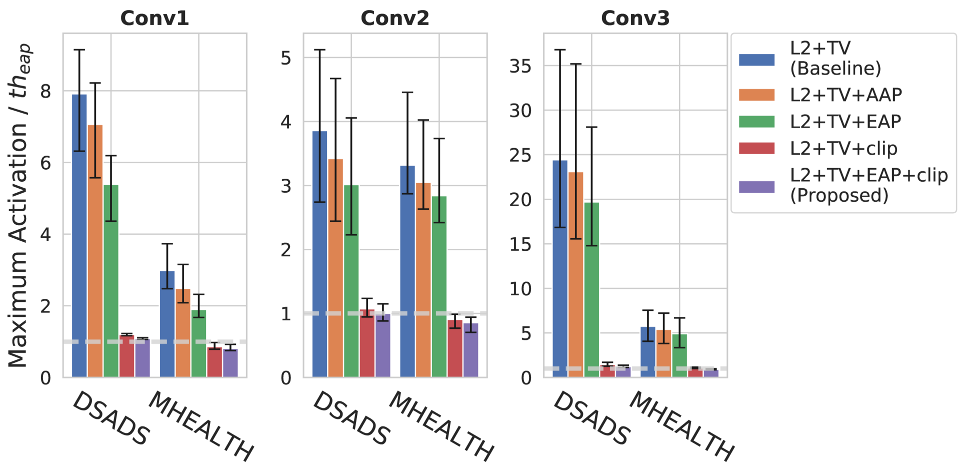


Figure 4.3: The ratio of the maximum activation of the generated signal to the threshold ($th_{eap}^{(l,u)}$). The values in the figure are averages in each layer. The error bars show the minimum and maximum values over 5 runs. LPF was excluded because the number of failed unit is too large.

shows the 95th percentile of the distribution of activation values in the training data, and these activation values are two times larger than the 95th percentile. The extreme activation values (and the corresponding signals) are assumed to be significantly different from those of the training data. In contrast, when EAP is used, the activation values are suppressed and have reasonable values, which demonstrates the effectiveness of EAP.

We can confirm the effectiveness of the activation clipping from Fig. 4.6. As shown in the left Fig. (L2+TV+EAP), even when EAP was used, we could not suppress abnormal activation values in this case. This is because EAP cannot suppress an individual outlying activation value since it penalizes based on a representative value of $\boldsymbol{X}^{(l,u)}$. In contrast, introducing activation clipping makes it possible to prevent signals with extremely large activation values from becoming larger, resulting in the generation of smoother signals.

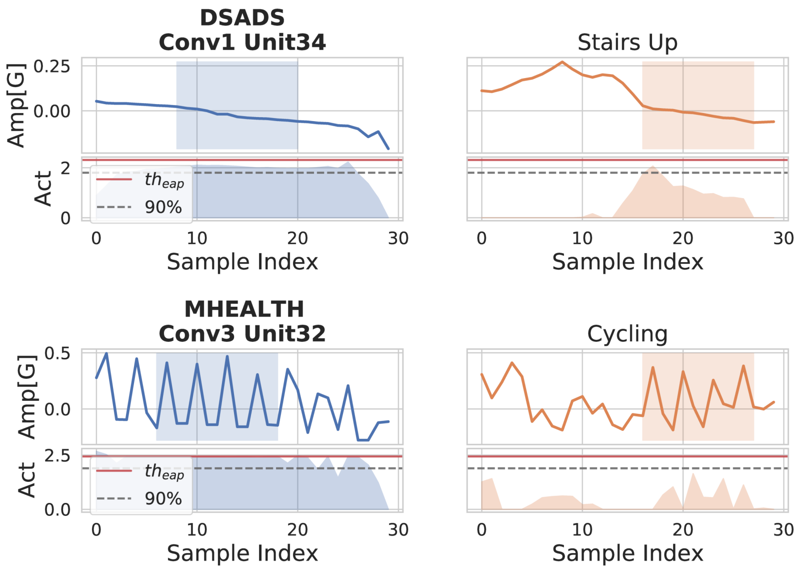


Figure 4.4: Signals generated by L2+TV+EAP+clip (blue signals) and their most similar training segments (orange signals); the corresponding activation values are shown in the lower panels. “Amp” and “Act” represent “amplitude” and “activation,” respectively. These pairs are selected based on the MSEs ($d(u, i)$).

4.6 Activation Atlas

Here, we describe applications of the AM method and their effectiveness in the development and analysis of human activity recognition neural networks using publicly available datasets. Because a network has a large number of hidden units, it is not efficient to browse the generated signals from each unit. Therefore, to facilitate the analysis, we developed an “activation atlas” [12] for activity recognition. An activation atlas is a map of the visual features of an object recognition neural network. Here, we generate a 2D map of human activity features by transforming signals generated from each unit using our method (i.e., a 30-dimensional data point) into a 2D data point using t-SNE [46]. The application enables two types of analysis: (i) analysis of automatically extracted feature representations, and (ii) parameter selection.

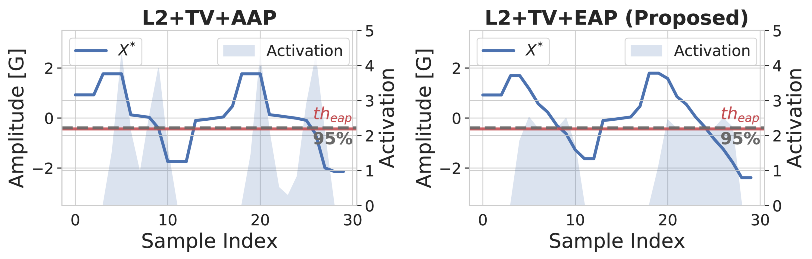


Figure 4.5: Examples of generated signals by methods with and without EAP for the same unit (Conv1 Unit03 of MHEALTH model).

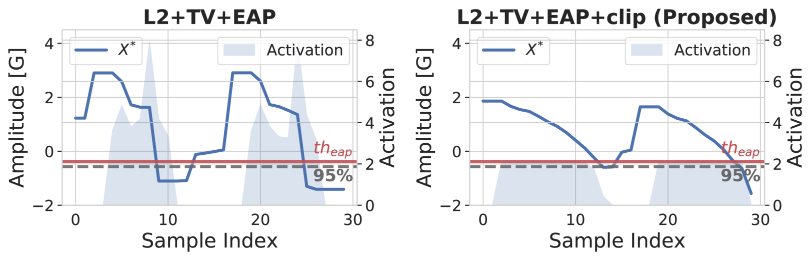
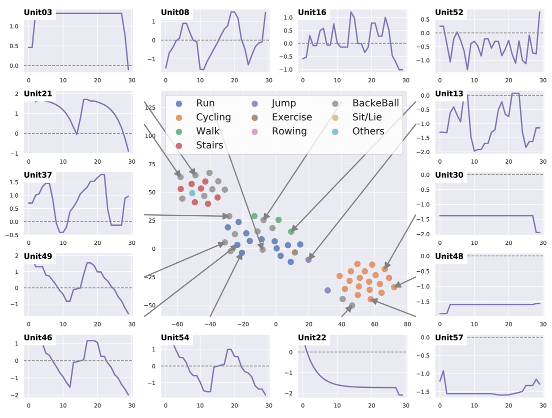


Figure 4.6: Examples of signals generated by methods with and without clipping for the same unit (Conv1 Unit02 of DSADS model).

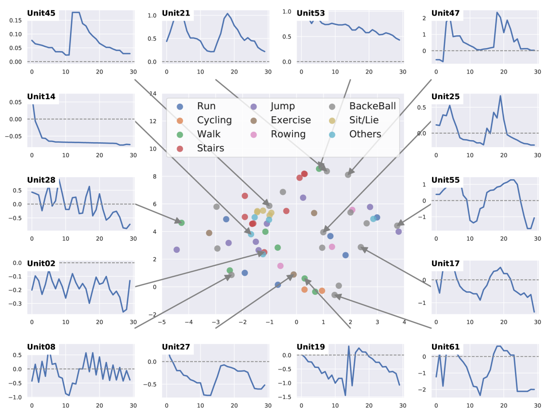
4.6.1 Analysis of Extracted Feature Representations

One of the advantages of using neural networks is automated feature engineering. The atlas enables us to easily browse the distribution of feature representations extracted in different units and layers. Fig. 4.7 shows the atlas and examples of generated signals from Conv1 (left) and Conv3 (right) in the model trained on DSADS. Each point corresponds to a unit (and generated signals), and an identifier of the unit is associated with the signals. In addition, the color of each point corresponds to the activity label of the training segment that is most similar to the generated signals of interest.

As shown in the left figure, the first layer of the network seems to extract simple features that look like slopes or flat waveforms. We can find three clusters on the map. The cluster at the top-left and bottom-right are consisting of the signals with positive and negative values, respectively. Although the value ranges are different, many simple signals with similar shapes are extracted in the first layer. In contrast, as shown in the right figure, data points of the third layer are scattered over the 2D map. Waveforms with jagged high-frequency components combined with simple slopes, e.g., Unit25 and



(1) Conv1 (1st layer)



(2) Conv3 (3rd layer)

Figure 4.7: Activation Atlas for human activities generated from the 1st and 3rd layers trained on DSADS. The center of each figure shows the Activation Atlas. A data point corresponding to a unit is colored according to the class of the most similar signals. Waveforms surrounding the map show signals generated by the units. The horizontal and vertical axes of the waveform graph represent the sample index and the amplitude [G], respectively.

Unit28, and waveforms with repeated complex patterns, e.g., Unit02, were generated.

Although deeper layers in an activity recognition network have been considered to extract more complex patterns than shallow layers [41], our study is the first to confirm this fact by actually generating signals from these layers of activity recognition networks.

As mentioned above, our method permits us to understand how the network was trained in each layer.

4.6.2 Parameter Selection

Using smaller neural networks is effective for activity recognition on a mobile device with limited computation power and battery life. Here, we verify the usefulness of analyzing the distribution of extracted feature representations using AM to determine the size of a neural network while maintaining recognition accuracy.

We trained activity recognition models on the MHEALTH dataset with a different number of units (convolutional filters), and the models were trained five times with different random seeds for each setting. Fig. 4.8 (1)-(6) show the AM results (atlas) for the model obtained in the first training and Fig. 4.8 (7) shows the averaged recognition accuracy (F1-measure). The error bars in Fig. 4.8 (7) are the maximum and minimum recognition accuracies in the five runs. As shown in Fig. 4.8 (7), the recognition accuracy is close to the upper limit when the number of units is 16. When unit=8, few features are extracted from the lower left region. In contrast, when unit=16, several units could cover the overall area of the lower left region, which is considered to be the reason why this setting achieved the upper limit. By visualizing such signals in the distributions of Units=32, 64, and 128, we can confirm whether these signals are required for recognition.

Interestingly, the points of the network with four units are located at the cluster centers of the points of the network with 256 units. Although the F-measure of the network with four units is poor, Fig. 4.8 shows that the network with four units attempts to learn features of important signals to maximize the overall accuracy of the dataset.

As mentioned above, our method provides insights related to hyperparameter selection and permits us to reveal interesting behaviors of human activity recognition networks.

4.7 Discussion

In this study, we assumed that the distribution of the activation values of each unit was one-sided normal distribution, and the value greater than approximate 95% point was considered as an outlier using the 2-sigma rule. However, the criterion for judging an outlier varies from situation to situation, and 2-sigma rule used in this study is just one option to define the threshold. For example, if the unit responds only to very special signals, it may be possible to detect an outlier at the 99% point estimated from the training data. The method proposed in this study can be expected to prompt the optimization around the threshold value and can be applied to other threshold definitions according to the assumptions. However, there is a possibility that the output of a unit may have a distribution that is different from the normal distribution, such as a Gaussian mixture distribution. In such cases, we cannot assure the effectiveness of our proposed method,

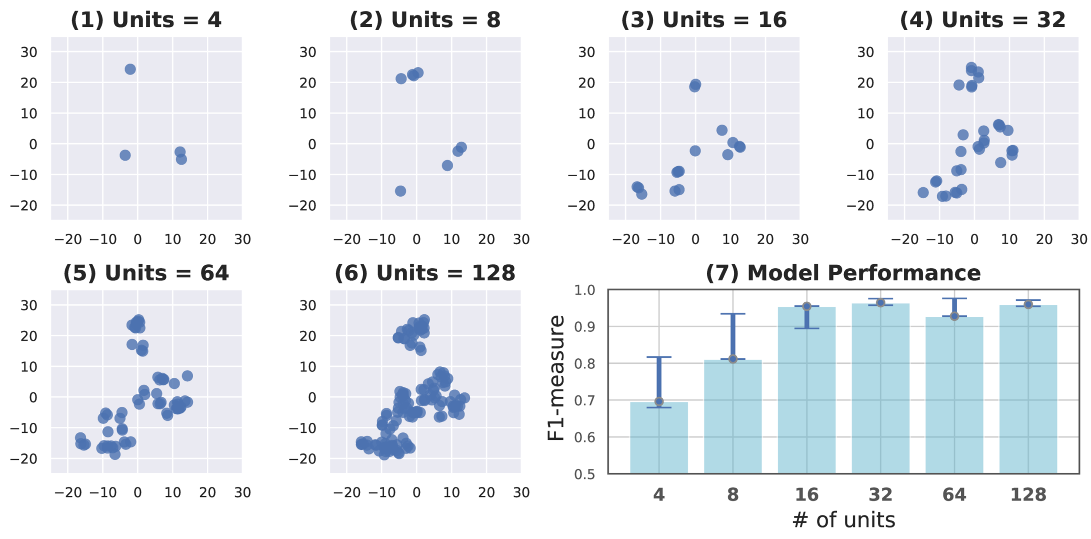


Figure 4.8: 2D maps of neural networks with different unit numbers. Each network is composed of three convolutional layers, with each of the layers constituting n units ($n = 4, 8, 16, 32, 64, 128$), and has the SF-HF architecture. The number of units in the fully connected layers is 256. The signals are generated from the third layer. Each network was trained on MHEALTH.

which uses a single scalar value as a threshold value for anomaly detection. Further investigation is needed to determine in which cases our proposed method is not effective.

4.8 Conclusion

This study revealed the feature extraction functions of human activity recognition neural networks using AM. To generate signals that maximize the activation of a convolutional unit, we proposed new regularization techniques leveraging the distribution of activation values from training data. We quantitatively and qualitatively evaluated the effectiveness of the proposed techniques using publicly available datasets with the 2-sigma rule as a method of determining the threshold. As a part of our future work, we would like to investigate whether the proposed method is also effective in the other ways of determining threshold values.

Chapter 5

Conclusion and Future Work

5.1 Thesis Summary

In this thesis, we developed three methods for recognizing workers' activities using wearable sensors. We believe that these methods can contribute to achieve smart manufacturing and logistics by incorporating information about real-world human works into the cyberspace. Specifically, we proposed a new work activity recognition model, a new dataset for work activity recognition, and an analysis technique for activity recognition models.

To summarize this study, we reflect on the research goals described in Chapter 1 and summarize the achievements here.

- **Recognize Work Operations of Workers with a Limited Amount of Training Data:** In a factory production line, the work assigned to each worker is different. In addition, the work procedures may differ depending on the skill level of the worker and mistakes. Therefore, it is difficult to use labeled training data collected from other workers as training data for a target worker. Therefore, it is necessary to construct a work activity recognition model with limited training data collected from each worker.
  In Chapter.2, we proposed LOS-Net, a work activity recognition model that efficiently extracts only necessary features with a lightweight model. To realize the concept of LOS-Net, three components were proposed. The WPCP module extracts the short-term and long-term context of work activities with relatively few

parameters by utilizing dilated convolution. The Boundary Detector estimates the start and end times of each work activity, which enables robust boundary detection against changes in the work order. The Refinement Module corrects errors in the decoder output by utilizing prior knowledge such as the order of operations.

- **New Large-scale Multimodal Dataset in Industrial Domain:** Many IoT devices are installed in industrial sites, and the usage log of IoT devices strongly relates to the user's activities at the time the data was generated. Therefore, it can be a useful resource for activity recognition. However, there is no publicly available dataset for work activity recognition that includes both sensor data and data from IoT devices. Furthermore, compared to sensor data, IoT device data is very sparse in time, and how to fuse effectively them has not been fully explored. In Chapter 3, we constructed a large-scale multimodal work activity recognition dataset in the industrial domain called the OpenPack dataset, in which 16 subjects' work activities during packaging were recorded for 53 hours with 9 different sensors. In addition to sensor data, we included data from IoT devices and metadata about subjects and orders. We proposed a novel activity recognition model for sensor and IoT data, called a ladder-shaped two-stream network (LTS-Net). LTS-Net processes sensor data and high-confidence IoT data in different streams and updates information in each stream by referring to information in different streams while preserving information on the IoT data in deeper layers.
  We proposed LOS-Net in Chapter 2, under the realistic assumption where labeled training data is limited. In contrast, LTS-Net was designed under the assumption where there is a large amount of labeled training data including data from IoT devices, in order to take advantage of the size and modality of the OpenPack dataset. Therefore, LTS-Net outperformed LOS-Net in the evaluation of Chapter 3. Note that, work process context pooling (WPCP) module was used as the feature extractor in LTS-Net, suggesting that the module proposed for LOS-Net is also effective under other assumptions.

- **Visualization of the Concept Learned by a Unit of the Neural Network:** Many of SOTA activity recognition methods are based on neural networks. In order to optimize manufacturing processes by utilizing the results of activity recognition, it is necessary to obtain managers' understanding and trust of the activity recognition algorithm. However, few managers are familiar with ICT technologies and neural networks, and it is difficult for them to trust the results derived from incomprehensible algorithms, which may be a potential bottleneck to the introduction of work activity recognition technologies into the site. Therefore, there is a need for a method to explain the behavior of activity recognition neural networks to the managers.

In Chapter 4, we proposed a visualization method to reveal the features extracted by a trained activity recognition neural network. Activation maximization is one of the visualization methods which visualizes the features extracted by the unit of interest by generating the input signal to which the unit of interest responds most strongly. In this study, we applied activation maximization to activity recognition neural networks. In generating the input signal, the design of the regularization term is important. We proposed a regularization method that determines whether the input signal (acceleration) is abnormal based on its activation value. Based on the proposed method, we created an activation atlas, which provides a bird's-eye view of the distribution of features extracted by the recognition model.

In Chapter 2 and Chapter 3, we proposed the work activity recognition models based on deep learning. There are a lot of managers who are not familiar with ICT. We believe that the proposed visualization techniques including an activation atlas can deepen their understanding and gain their trust in the deep learning-based models by visualizing a part of the decision-making process. This will lower the barrier for installing work activity recognition models on the site and lead to their continuous use.

5.2 FutureWork

In this thesis, we have proposed a solution to the problem of developing a work activity recognition technology for practical use. In the following, we summarize the advanced topics that should be addressed in the future.

5.2.1 Robustness of Work Activity Recognition Models for Unexpected Worker's Activity

In this thesis, we proposed a work activity recognition model and confirmed that it is capable of recognizing work activities with high accuracy. In order to further improve the recognition performance, it is necessary to deal with unexpected activities. The proposed model utilizes prior knowledge of work activities, and we have qualitatively confirmed that recognition becomes difficult when unexpected actions are taken. Although it does not happen frequently, there are cases such as mistakes made during work, or activities not related to the work, such as stopping work to talk to another worker or manager. To deal with this problem, it is necessary not only to recognize actions, but also to explore tasks to detect irregular or wrong actions, and to use the results to increase robustness against unexpected actions.

5.2.2 Developping Algorithm for Bottleneck Assessment and Process Optimization

In this study, we focused on understanding the work activities of factory workers to solve this problem. However, we have not yet sufficiently studied algorithms for optimization of the entire factory, including robots. Using the work activity data generated by the activity recognition technology, an algorithm is needed to detect where in the production line there is inefficiency and which work activities are prone to procedural errors.

Acknowledgment

I would like to thank many people who have supported my research for the past six years in Hara Laboratory.

Especially, I would like to express my deepest gratitude to Associate Professor Takuya Maekawa, who gave me his guidance and generous advice in the course of my research. Working under the guidance of Professor Maekawa, I was able to learn the excitement of research and had the valuable experience of gradually realizing what seemed impossible at first, thanks to his advice. In preparing this thesis, he gave me a great deal of advice on the structure of the paper as well as on the detailed expressions. Without his great patience and persistence, I would not have been able to complete this thesis.

I am also grateful to Professor Takahiro Hara for sharing his time and expertise, and for his constructive advice on my research. He also provided me with many opportunities such as ACT-X, which gave me many valuable opportunities. I believe that these experiences and the relationships I made through them will be a valuable asset in the future.

I would like to express my sincere appreciation to the committee members for my thesis. Dr. Yasuyuki Matsushita and Dr. Toru Fujiwara provided many insightful and constructive comments that considerably improved the quality of my thesis.

I would also like to express my sincere appreciation to Dr. Daichi Amakata, Dr. Atsuo Inomata, Dr. Naoto Yanai, Dr. Makoto Onizuka, Dr. Yuki Arase, Dr. Chuan XIAO, Dr. Fumio Okura, Dr. Shinji Shimojo, Dr. Susumu Date, Dr. Norihiro Hagita, and Dr. Takahiro Miyashita, Dr. Satoru Satake, for their sincere advice and support on my research during my mid-term defenses, and pre-defense.

In particular, I would like to thank Teerawat Kumrai, Xia Qingxin, Dissanayake Thilina Madushan, Jaime Morales, Zhi Li, Heng Zhou, Yuchen Ji, for providing valuable discussions and ideas and for sharing their research experiences with me.

This paper was supported by Toshiba Corporation, Corporate Manufacturing Engineering Center. In particular, I am grateful to Dr.Yasuo Namioka and Atsushi Wada for providing me with various research hints, such as issues in actual manufacturing sites. I also thank Chikako Kawabe, Kana Yasuda, Makiko Higashinaka, Makiko Otsuka, and others for their support in labeling data and many other ways. Without their contributions, my research would not have been possible. Thank you very much.

Last but not least, I would like to express my deepest gratitude to my family. Without their support, this thesis would not have been possible. I would like to thank my parents for supporting me throughout my efforts and adventures.

REFERENCE

[1] M. Aehnelt, E. Gutzeit, and B. Urban. Using activity recognition for the tracking of assembly processes: Challenges and requirements. *WOAR*, 2014:12–21, 2014.

[2] M. Al-Amin, R. Qin, W. Tao, D. Doell, R. Lingard, Z. Yin, and M. C. Leu. Fusing and refining convolutional neural network models for assembly action recognition in smart manufacturing. *Proceedings of the Institution of Mechanical Engineers*, 2020.

[3] M. Al-Amin, W. Tao, D. Doell, R. Lingard, Z. Yin, M. C. Leu, and R. Qin. Action recognition in manufacturing assembly using multimodal sensor fusion. *Procedia Manufacturing*, 39:158–167, 2019.

[4] S. S. Alia, K. Adachi, N. Nahid, H. Kaneko, P. Lago, and S. Inoue. Bento packaging activity recognition challenge, 2021.

[5] S. S. Alia, P. Lago, S. Takeda, K. Adachi, B. Benaissa, M. A. R. Ahad, and S. Inoue. Summary of the cooking activity recognition challenge. In *Human Activity Recognition Challenge*, pages 1–13. 2021.

[6] A. Araujo, W. Norris, and J. Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019.

[7] O. Banos, R. Garcia, J. A. Holgado-Terriza, M. Damas, H. Pomares, I. Rojas, A. Saez, and C. Villalonga. mHealthDroid: a novel framework for agile development of mobile health applications. In *Proceedings of the International Workshop on Ambient Assisted Living*, pages 91–98, 2014.

[8] O. Banos, M. Toth, M. Damas, H. Pomares, and I. Rojas. Dealing with the effects of sensor displacement in wearable activity recognition. *Sensors*, 14(6):9995–10023, 2014.

[9] B. Barshan and M. C. Yüksek. Recognizing daily and sports activities in two open source machine learning environments using body-worn sensor units. *The Computer Journal*, 57(11):1649–1667, 2014.

[10] Y. Ben-Shabat, X. Yu, F. Saleh, D. Campbell, C. Rodriguez-Opazo, H. Li, and S. Gould. The ikea asm dataset: Understanding people assembling furniture through actions, objects and pose. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 847–859, 2021.

[11] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[12] S. Carter, Z. Armstrong, L. Schubert, I. Johnson, and C. Olah. Activation atlas. *Distill*, 2019.

[13] C. Chen, R. Jafari, and N. Kehtarnavaz. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *Proceedings of the IEEE International Conference on Image Processing*, pages 168–172. IEEE, 2015.

[14] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017.

[15] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 801–818, 2018.

[16] M. Dallel, V. Havard, D. Baudry, and X. Savatier. Inhard-industrial human action recognition dataset in the context of industrial collaborative robotics. In *Proceed-*

ings of the 2020 IEEE International Conference on Human-Machine Systems, pages 1–6. IEEE, 2020.

[17] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision*, 2018.

[18] T. Dissanayake, T. Maekawa, D. Amagata, and T. Hara. Detecting door events using a smartphone via active sound sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2(4):1–26, 2018.

[19] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[20] S. Feldhorst, M. Masoudenijad, M. ten Hompel, and G. A. Fink. Motion classification for analyzing the order picking process using mobile sensors. In *Procceedings of International Conference of Pattern Recognition Applications and Methods*, pages 706–713, 2016.

[21] A. R. M. Forkan, F. Montori, D. Georgakopoulos, P. P. Jayaraman, A. Yavari, and A. Morshed. An industrial iot solution for evaluating workers' performance via activity recognition. In *Proceedings of the 39 th IEEE International Conference on Distributed Computing Systems*, pages 1393–1403, 2019.

[22] M. Fukui, G. Kokubun, and T. Nozaki. Visualization of important human motion feature using convolutional neural network. In *Proceedings of the IEEE International Conference on Industrial Technology*, pages 406–411. IEEE, 2020.

[23] C. E. Galván-Tejada, F. E. López-Monteagudo, O. Alonso-González, J. I. Galván-Tejada, J. M. Celaya-Padilla, H. Gamboa-Rosales, R. Magallanes-Quintanar, and L. A. Zanella-Calzada. A generalized model for indoor location estimation using environmental sound from human activity recognition. *ISPRS International Journal of Geo-Information*, 7(3):81, 2018.

[24] E. Garcia-Ceja, C. E. Galván-Tejada, and R. Brena. Multi-view stacking for activity recognition with sound and accelerometer data. *Information Fusion*, 40:45–56, 2018.

[25] S. Gashi, E. Di Lascio, and S. Santini. Using unobtrusive wearable sensors to measure the physiological synchrony between presenters and audience members. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(1):1–19, 2019.

[26] N. Y. Hammerla, S. Halloran, and T. Ploetz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *CoRR*, abs/1604.08880, 2016.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[28] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In *Wavelets*, pages 286–297. 1990.

[29] S. Inoue, P. Lago, T. Hossain, T. Mairittha, and N. Mairittha. Integrating activity recognition and nursing care records: The system, deployment, and a verification study. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(3), 2019.

[30] M. M. Islam and T. Iqbal. Multi-gat: A graphical attention-based hierarchical multimodal representation learning approach for human activity recognition. *IEEE Robotics and Automation Letters*, 6(2):1729–1736, 2021.

[31] M. M. Islam and T. Iqbal. Mumu: Cooperative multitask learning-based guided multimodal fusion. AAAI, 2022.

[32] Y.-W. Kim, W.-H. Cho, K.-S. Kim, and S. Lee. Inertial-measurement-unit-based novel human activity recognition algorithm using conformer. *IEEE Sensors*, 22(10):3932, 2022.

[33] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[34] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *Proceedings of the the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9404–9413, 2019.

[35] M. Klumpp, M. Hesenius, O. Meyer, C. Ruiner, and V. Gruhn. Production logistics and human-computer interaction—state-of-the-art, challenges and requirements for the future. *The International Journal of Advanced Manufacturing Technology*, 105(9):3691–3709, 2019.

[36] M. Klumpp, M. Hesenius, O. Meyer, C. Ruiner, and V. Gruhn. Production logistics and human-computer interaction—state-of-the-art, challenges and requirements for the future. *The International Journal of Advanced Manufacturing Technology*, 105(9):3691–3709, 2019.

[37] Q. Kong, Z. Wu, Z. Deng, M. Klinkigt, B. Tong, and T. Murakami. Mmact: A large-scale dataset for cross modal human action understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, October 2019.

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

[39] P. Lago, S. Takeda, K. Adachi, S. S. Alia, M. Matsuki, B. Benai, S. Inoue, and F. Charpillet. Cooking activity dataset with macro and micro activities, 2020.

[40] Q. V. Le, M. Ranzato, R. Monga, M. Devin, G. Corrado, K. Chen, J. Dean, and A. Y. Ng. Building high-level features using large scale unsupervised learning. In *Proceedings of the International Conference on Machine Learning*, 2012.

[41] X. Li, X. Si, L. Nie, J. Li, R. Ding, D. Zhan, and D. Chu. Understanding and improving deep neural network for activity recognition. *CoRR*, abs/1805.07020, 2018.

[42] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[43] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[44] C. Lübbe, B. Friedrich, S. Fudickar, S. Hellmers, and A. Hein. Feature based random forest nurse care activity recognition using accelerometer data. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 408–413, 2020.

[45] H. Ma, W. Li, X. Zhang, S. Gao, and S. Lu. Attnsense: Multi-level attention mechanism for multimodal human activity recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 3109–3115, 2019.

[46] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[47] T. Maekawa, Y. Kishino, Y. Yanagisawa, and Y. Sakurai. Mimic sensors: battery-shaped sensor node for detecting electrical events of handheld devices. In *Proceedings of the International Conference on Pervasive Computing*, pages 20–38, 2012.

[48] A. Mahendran and A. Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5188–5196, 2015.

[49] R. Michel. 2016 warehouse/dc operations survey: Ready to confront complexity, Nov 2016.

[50] I. Mohino-Herranz, R. Gil-Pita, M. Rosa-Zurera, and F. Seoane. Activity recognition using wearable physiological measurements: Selection of features from a comprehensive literature study. *Sensors*, 19(24):5524, 2019.

[51] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[52] J. Morales, N. Yoshimura, Q. Xia, A. Wada, Y. Namioka, and T. Maekawa. Acceleration-based human activity recognition of packaging tasks using motif-guided attention networks. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications*, pages 1–12. IEEE, 2022.

[53] F. Moya Rueda, R. Grzeszick, G. A. Fink, S. Feldhorst, and M. Ten Hompel. Convolutional neural networks for human activity recognition using body-worn sensors. In *Informatics*, volume 5, page 26. Multidisciplinary Digital Publishing Institute, 2018.

[54] S. Münzner, P. Schmidt, A. Reiss, M. Hanselmann, R. Stiefelhagen, and R. Dürichen. CNN-based sensor fusion techniques for multimodal human activity recognition. In *Proceedings of the ACM International Symposium on Wearable Computers*, pages 158–165, 2017.

[55] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski. Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4467–4477, 2017.

[56] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 427–436, 2015.

[57] A. M. Nguyen, J. Yosinski, and J. Clune. Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *CoRR*, abs/1602.03616, 2016.

[58] F. Niemann, S. Lüdtke, C. Bartelt, and M. Ten Hompel. Context-aware human activity recognition in industrial processes. *Sensors*, 22(1):134, 2022.

[59] F. Niemann, C. Reining, F. Moya Rueda, N. R. Nair, J. A. Steffens, G. A. Fink, and M. Ten Hompel. Lara: Creating a dataset for human activity recognition in logistics using semantic attributes. *Sensors*, 20(15):4083, 2020.

[60] F. M. Noori, E. Garcia-Ceja, M. Z. Uddin, M. Riegler, and J. Tørresen. Fusion of multiple representations extracted from a single sensor's data for activity recognition using CNNs. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–6, 2019.

[61] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Berkeley mhad: A comprehensive multimodal human action database. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pages 53–60. IEEE, 2013.

[62] K. Ohara, T. Maekawa, and Y. Matsushita. Detecting state changes of indoor everyday objects using wi-fi channel state information. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3), 2017.

[63] F. Ordóñez and D. Roggen. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[64] M. Perslev, M. H. Jensen, S. Darkner, P. J. Jennum, and C. Igel. U-time: A fully convolutional network for time series segmentation applied to sleep staging. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 4417–4428, 2019.

[65] A. Rahman, N. Nahid, I. Hassan, and M. A. R. Ahad. Nurse care activity recognition: Using random forest to handle imbalanced class problem. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 419–424, 2020.

[66] C. Reining, F. Niemann, F. Moya Rueda, G. A. Fink, and M. ten Hompel. Human activity recognition for production and logistics—a systematic literature review. *Information*, 10(8):245, 2019.

[67] C. Reining, M. Schlangen, L. Hissmann, M. ten Hompel, F. Moya, and G. A. Fink. Attribute representation for human activity recognition of manual order

picking activities. In *Proceedings of the 5th international Workshop on Sensor-based Activity Recognition and Interaction*, pages 1–10, 2018.

[68] A. Reiss and D. Stricker. Introducing a new benchmarked dataset for activity monitoring. In *Proceedings of the International Symposium on Wearable Computers*, pages 108–109, 2012.

[69] R. S. Renu. Unsupervised method of determining cycle times of manual assembly processes. In *Proceedings of the International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 83983, 2020.

[70] J.-L. Reyes-Ortiz, L. Oneto, A. Sama, X. Parra, and D. Anguita. Transition-aware human activity recognition using smartphones. *Neurocomputing*, 171:754–767, 2016.

[71] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, et al. Collecting complex activity datasets in highly rich networked sensor environments. In *Proceedings of the 7 th International Conference on Networked Sensing Systems*, pages 233–240, 2010.

[72] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. W. III, and A. F. Frangi, editors, *Proceedings of the Medical Image Computing and Computer-Assisted Intervention*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241, 2015.

[73] A. Saeed, T. Ozcelebi, and J. Lukkien. Multi-task self-supervised learning for human activity detection. *the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):61, 2019.

[74] S. Saks. *Theory of the integral. 2. ed. English translation by L. C. Young. With two additional notes by S. Banach.*, volume 7. 1937.

[75] S. S. M. Salehi, D. Erdogmus, and A. Gholipour. Tversky loss function for image segmentation using 3d fully convolutional deep networks. In *Proceedings of the*

International workshop on machine learning in medical imaging, pages 379–387. Springer, 2017.

[76] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673, 2016.

[77] D. Schlögl and H. Zsifkovits. Manuelle kommissioniersysteme und die rolle des menschen. *Berg Huettenmaenn Monatsh*, 161:225–228, 2016.

[78] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[79] F. Sener, D. Chatterjee, D. Shelepov, K. He, D. Singhania, R. Wang, and A. Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21096–21106, 2022.

[80] L. Shi, Y. Zhang, J. Cheng, and H. Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019.

[81] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Proceedings of the International Conference on Learning Representations*, 2014.

[82] S. P. Singh, M. K. Sharma, A. Lay-Ekuakille, D. Gangwar, and S. Gupta. Deep convlstm with self-attention for human activity decoding using wearable sensors. *IEEE Sensors*, 21(6):8575–8582, 2020.

[83] E. Soleimani and E. Nazerfard. Cross-subject transfer learning in human activity recognition systems using generative adversarial networks. *CoRR*, abs/1903.12489, 2019.

[84] E. H. Spriggs, F. De La Torre, and M. Hebert. Temporal segmentation and activity classification from first-person sensing. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 17–24. IEEE, 2009.

[85] S. Stein and S. J. McKenna. Combining embedded accelerometers with computer vision for recognizing food preparation activities. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 729–738, 2013.

[86] T. Stiefmeier, D. Roggen, and G. Troster. Fusion of string-matched templates for continuous activity recognition. In *Procceedings of the 11th IEEE International Symposium on Wearable Computers*, pages 41–44, 2007.

[87] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *CoRR*, abs/1609.03499, 2016.

[88] A. Van Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of the International Conference on Machine Learning*, pages 1747–1756, 2016.

[89] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of the Advances in neural information processing systems*, pages 5998–6008, 2017.

[90] J. Wang, Y. Chen, Y. Gu, Y. Xiao, and H. Pan. Sensorygans: an effective generative adversarial framework for sensor-based human activity recognition. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–8. IEEE, 2018.

[91] J. Wang, V. W. Zheng, Y. Chen, and M. Huang. Deep transfer learning for cross-domain activity recognition. In *Proceedings of the 3rd International Conference on Crowd Science and Engineering*, pages 1–8, 2018.

[92] J. C. Y. Wong, J. Wang, E. Y. Fu, H. V. Leong, and G. Ngai. Activity recognition and stress detection via wristband. In *Proceedings of the 17th International*

Conference on Advances in Mobile Computing & Multimedia, pages 102–106, 2019.

[93] R. Xi, M. Hou, M. Fu, H. Qu, and D. Liu. Deep dilated convolution on multimodality time series for human activity recognition. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–8, 2018.

[94] Q. Xia, J. Korpela, Y. Namioka, and T. Maekawa. Robust unsupervised factory activity recognition with body-worn accelerometer using temporal structure of multiple sensor data motifs. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(3):1–30, 2020.

[95] Q. Xia, A. Wada, J. Korpela, T. Maekawa, and Y. Namioka. Unsupervised factory activity recognition with wearable sensors using process instruction information. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(2):1–23, 2019.

[96] S. Yan, X. Xiong, A. Arnab, Z. Lu, M. Zhang, C. Sun, and C. Schmid. Multiview transformers for video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3333–3343, 2022.

[97] S. Yan, Y. Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the Thirty-second AAAI Conference on Artificial Intelligence*, 2018.

[98] R. Yao, G. Lin, Q. Shi, and D. C. Ranasinghe. Efficient dense labelling of human activity sequences from wearables using fully convolutional networks. *Pattern Recognition*, 78:252–266, 2018.

[99] V. Yavas and Y. D. Ozkan-Ozen. Logistics centers in the new industrial era: A proposed framework for logistics center 4.0. *Transportation Research Part E: Logistics and Transportation Review*, 135, mar 2020.

[100] V. Yavas and Y. D. Ozkan-Ozen. Logistics centers in the new industrial era: A proposed framework for logistics center 4.0. *Transportation Research Part E: Logistics and Transportation Review*, 135:101864, 2020.

[101] N. Yoshimura, T. Maekawa, and T. Hara. Preliminary investigation of visualizing human activity recognition neural network. In *Proceedings of the International Conference on Mobile Computing and Ubiquitous Network*, pages 1–2. IEEE, 2019.

[102] N. Yoshimura, T. Maekawa, and T. Hara. Toward understanding acceleration-based activity recognition neural networks with activation maximization. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

[103] N. Yoshimura, T. Maekawa, T. Hara, A. Wada, and Y. Namioka. Acceleration-based activity recognition of repetitive works with lightweight ordered-work segmentation network. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(2), 2022.

[104] N. Yoshimura, J. Morales, T. Maekawa, and T. Hara. OpenPack Dataset: A large-scale dataset for recognizing packaging works in IoT-enabled logistic environments. In *IPSJ SIG-DPS Technical Reports*, 2022.

[105] J. Yosinski, J. Clune, A. M. Nguyen, T. J. Fuchs, and H. Lipson. Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579, 2015.

[106] Z. Yu, C. Feng, M.-Y. Liu, and S. Ramalingam. Casenet: Deep category-aware semantic edge detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5964–5973, 2017.

[107] M. Zeng, H. Gao, T. Yu, O. J. Mengshoel, H. Langseth, I. Lane, and X. Liu. Understanding and improving recurrent networks for human activity recognition by continuous attention. In *Proceedings of the ACM International Symposium on Wearable Computers*, pages 56–63, 2018.

[108] L. Zhang, W. Zhang, and N. Japkowicz. Conditional-unet: A condition-aware deep model for coherent human activity recognition from wearables. In *Proceedings of the 25th International Conference on Pattern Recognition*, pages 5889–5896, 2021.

[109] Y. Zhang, Z. Zhang, Y. Zhang, J. Bao, Y. Zhang, and H. Deng. Human activity recognition based on motion sensor using u-net. *IEEE Access*, 7:75213–75226, 2019.