Human activity recognition in healthcare: challenges, approaches and applications

By

XIN(REX) LIU

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Xin Liu, Chair

. Yu-Jui Yvonne Wan

Jiawei Zhang

Committee in Charge

2024

Contents

Abstract

Human Activity Recognition in Healthcare: Challenges, Approaches and Applications

Human activity recognition (HAR) technology that analyzes data acquired from various types of sensing devices, including wearable sensors and vision sensors, is getting considerable attention in the field of Artificial Intelligence (AI) driven healthcare systems. Human activities can be used to provide remote healthcare solutions by identifying particular movements such as falls, gait, and breathing disorders. HAR healthcare system can allow people to live more independent lifestyles and still have the safety of being monitored if more direct care is needed. Thanks to the development of machine learning technology, many machine learning methods have been employed in human activity recognition systems in healthcare. However, this field still faces many technical challenges. Some challenges are shared with other pattern recognition fields, such as a limited number of labeled data, while other challenges are unique to sensor-based activity recognition in healthcare and require dedicated methods for real-life healthcare applications, such as data noise of sensor factors in the data collection process.

In this dissertation, we start with the challenges of healthcare-oriented HAR systems and summarize the challenge-related machine learning approaches. To overview HAR healthcare applications with wearable sensors, we cover essential components of designing HAR healthcare systems, including sensor factors (e.g., type, number, and placement location), AI model selection (e.g., classical machine learning models versus deep learning models), and feature engineering.

Next, we present a new healthcare application of HAR, that is, Early Mobility Activity (EMA) recognition for Intensive Care Unit (ICU) patients, to illustrate the system design of HAR applications for healthcare. We identify insensitive wearable sensor orientation features and propose a segment voting process to improve the model accuracy and stability.

We further apply the state-of-the-art vision sensor-based HAR approaches in healthcare. We present a healthcare system (BWCNN) to use eye blinks to communicate with the outside world for Amyotrophic Lateral Sclerosis(ALS) patients. The system uses a Convolutional Neural Network (CNN) to predict the eyes' state, which is used to find the blinking pattern.

Then, we propose a MASTAF that can quickly learn from a few examples efficiently to solve the limited number of video samples in real-life HAR applications, a common challenge shared with computer vision. MASTAF takes input from a general video spatial and temporal representation,e.g., using 2D CNN, 3D CNN, and video Transformer. Then, to make the most of such representations, we use self- and cross-attention models to highlight the critical spatio-temporal region to increase the inter-class distance and decrease the intra-class distance. Last, MASTAF applies a lightweight fusion network and the nearest neighbor classifier to classify each query video. We demonstrate that MASTAF improves the state-of-the-art performance on three few-shot HAR video benchmarks.

Last, we present Multimodal Masked Autoencoders-Based One-Shot Learning (Mu-MAE), which represents a significant advancement in the field of HAR using multimodal sensors. Addressing the challenges posed by labor-intensive data collection and reliance on external pretrained models, MU-MAE introduces a synchronized masking strategy tailored for wearable sensors, coupled with a multimodal masked autoencoder architecture. This innovative approach compels the networks to capture more meaningful spatiotemporal features, facilitating effective self-supervised pretraining without the need for additional data. Furthermore, MU-MAE leverages the representations extracted from multimodal masked autoencoders to enhance cross-attention fusion, which highlights critical spatiotemporal features across different modalities while emphasizing differences between activity classes. Through comprehensive evaluations on MMAct one-shot classification datasets, MU-MAE demonstrates superior performance, achieving up to an 80.17% accuracy for five-way one-shot multimodal classification, thus establishing itself as a state-of-the-art solution in HAR for healthcare applications.

Acknowledgments

I extend my heartfelt gratitude to Professor Xin Liu for providing me with the invaluable opportunity to pursue my passion in research. Her encouragement has allowed me to flourish as a researcher. I also want to express my sincere appreciation to Professor Yu-Jui Yvonne Wan for her guidance and invaluable feedback throughout the project. I am immensely grateful to my dissertation committee members at the University of California, Davis, particularly Professors Jiawei Zhang, for their insightful comments and guidance. Lastly, I want to thank my family for their unwavering support, which words cannot adequately convey.

CHAPTER 1

Introduction

During the past decade, sensor technology has developed significantly in multiple perspectives, including computational power, size, accuracy, and manufacturing costs [**1**]. These advancements enable a wide range of sensors to be integrated into smartphones and other portable devices to make them more practical. While video surveillance, one particular type of video sensor being increasingly popular, can offer better video quality, more straightforward setup, lower cost, and secure communication [**2**]. Therefore, an increasing number of applications utilizing camera systems for Human Activity Recognition in healthcare have been proposed recently.

HAR allows machines to analyze and comprehend human activities from input data sources, such as wearable sensors and video sensors [**3**]. HAR is applied in various healthcare systems that involve direct or indirect interaction between humans and smart devices. For instance, patients with obesity, diabetes, or cardiovascular diseases have to strictly follow a healthy, well-balanced diet and a regular exercise schedule [**4**]. Hence, tracking daily activities with HAR system is necessary to give real-time feedback to patients about their progress and provide up-to-date reports to clinicians. Similarly, patients with declined in mental ability or mental disorders must be monitored continuously to identify unusual actions in time. The HAR system can detect abnormal activities and provide remote doctors assistance, preventing unwanted consequences [**5**].

Many previous work has adopted machine learning methods in human activity recognition [**6**]. They rely highly on feature extraction techniques, including time-frequency transformation [**7**], statistical approaches [**8**], and symbolic representation [**9**]. However, the features extracted are often human engineered and based on heuristics. There was a need for systematical feature extraction approaches to effectively capture distinguishable features for human activities. In recent years, deep learning has embraced conspicuous prosperity in modeling high-level abstractions from data in many areas such as computer vision, natural language processing, and speech processing [**10**]. After early work [**5**, **11**] examined the effectiveness of deep learning in human activity recognition,

related studies sprung up in HAR in healthcare. However, deep learning is still confronted with reluctant acceptance by healthcare researchers owing to medical data sparsity and great effort for data annotation. Therefore, in the Healthcare HAR problem, there is a great potential to improve the system performance with a limited number of labeled samples.

In this dissertation, we highlight the challenges of healthcare-oriented HAR systems and summarize the challenge-related machine learning approaches. The key contributions of this work are as follows:

- **An Overview of Human Activity Recognition Using Wearable Sensors: Healthcare and Artificial Intelligence** With the rapid development of the internet of things (IoT) and artificial intelligence (AI) technologies, human activity recognition has been applied in a variety of domains such as security and surveillance, human-robot interaction, and entertainment. Even though a number of surveys and review papers have been published, there is a lack of HAR overview papers focusing on healthcare applications that use wearable sensors. Therefore, we start with an overview of HAR applications in healthcare. We cover essential components of designing HAR systems including sensor factors (e.g., type, number, and placement location), AI model selection (e.g., classical machine learning models versus deep learning models), and feature engineering. In addition, we highlight the challenges of such healthcare-oriented HAR systems and propose several research opportunities for both the medical and the computer science community.
- **Early Mobility Activity Recognition for Intensive Care Unit Patients Using Accelerometers** In this work, we target a new healthcare application of HAR, that is, Early Mobility Activity (EMA) recognition for Intensive Care Unit (ICU) patients. EMA is essential for ICU patients who suffer from long-time immobilization. We propose an accurate AI model to recognize patients' EMA. To improve the model accuracy and stability, we identify features that are insensitive to sensor orientations, and propose a segment voting process that leverages a majority voting strategy to recognize each segment's activity. Our extensive results show that our algorithm improves the model accuracy from 77.78% to 81.86%, and reduces the model instability (standard deviation) from 16.69% to 6.92%,

comparing to the same AI model without our feature engineering and segment voting process.

- **BWCNN: Blink to Word, a Real-Time Convolutional Neural Network Approach** Amyotrophic lateral sclerosis or ALS is a progressive neurodegenerative disease that affects nerve cells in the brain and the spinal cord. While an individual affected by this disease loses all motor functions, they can still blink their eyes. We present a system (BWCNN) to use the eye blinks as a mode of communication with the outside world. The system uses a Convolutional Neural Network (CNN) to predict the state of the eyes and this state is used to find the blinking pattern. Once the pattern is obtained, it is mapped with a collection of phrases with a label for each pattern. Several state-of-the-art ConvNet architectures such as ResNet, SqueezeNet, DenseNet, and InceptionV3 were implemented in this system to evaluate their performance in order to choose the best one for eye state recognition. By tuning the hyperparameters of the networks such as batch size and the number of iterations, we analyzed the performance of different architectures based on the accuracy and the time taken for prediction. The goal of our system is to strike a balance between accuracy and the latency. High accuracy makes our prediction better while small latency makes it faster. Although the highest accuracy was obtained from ResNet (99.26%, 117ms latency). We found that InceptionV3 architecture struck the best balance for our system with an accuracy of 99.20%, and 94ms latency.
- **MASTAF: A Model-Agnostic Spatio-Temporal Attention Fusion Network for Few-shot Video Classification** We propose MASTAF, a Model-Agnostic Spatio-Temporal Attention Fusion network for few-shot video classification, to solve the limited number of video samples in real-life HAR applications, a common challenge shared with computer vision. MASTAF takes input from a general video spatial and temporal representation,e.g., using 2D CNN, 3D CNN, and video Transformer. Then, to make the most of such representations, we use self- and cross-attention models to highlight the critical spatio-temporal region to increase the inter-class distance and decrease the intra-class distance. Last, MASTAF applies a lightweight fusion network and a nearest neighbor classifier to classify each query video. We demonstrate that MASTAF improves the state-of-the-art

performance on three few-shot video classification benchmarks(UCF101, HMDB51, and Something-Something-V2), e.g., by up to 91.6%, 69.5%, and 60.7% for five-way one-shot video classification, respectively.

- **MU-MAE: Multimodal Masked Autoencoders-Based One-Shot Learning** Accurately recognizing human activities with multimodal sensors faces challenges due to the labor-intensive nature of data collection and annotation, and reliance on external pretrained models or additional data. To address these challenges, we introduce Multimodal Masked Autoencoders-Based One-Shot Learning (Mu-MAE). Mu-MAE integrates a multimodal masked autoencoder with a synchronized masking strategy tailored for wearable sensors. This masking strategy compels the networks to capture more meaningful spatiotemporal features, which enables effective self-supervised pretraining without the need for external data. Furthermore, Mu-MAE leverages the representation extracted from multimodal masked autoencoders as prior information input to a cross-attention multimodal fusion layer. This fusion layer emphasizes spatiotemporal features requiring attention across different modalities while highlighting differences from other classes, aiding in the classification of various classes in metric-based one-shot learning. Comprehensive evaluations on MMAct one-shot classification show that Mu-MAE outperforms all the evaluated approaches, achieving up to an 80.17% accuracy for five-way one-shot multimodal classification, without the use of additional data.

CHAPTER 2

An Overview of Human Activity Recognition Using Wearable Sensors

With the rapid development of the internet of things (IoT) and artificial intelligence (AI) technologies, human activity recognition has been applied in a variety of domains such as security and surveillance, human-robot interaction, and entertainment. Even though a number of surveys and review papers have been published, there is a lack of HAR overview papers focusing on healthcare applications that use wearable sensors. Therefore, we fill in the gap in this work. We cover essential components of designing HAR systems including sensor factors (e.g., type, number, and placement location), AI model selection (e.g., classical machine learning models versus deep learning models), and feature engineering. In addition, we highlight the challenges of such healthcare-oriented HAR systems and propose several research opportunities for both the medical and the computer science community.

2.1. Introduction

Human activity recognition has been actively researched in the past decade, thanks to the increasing number of deployed smart devices such as smartphones and IoT devices. Based on the type of data being processed, a HAR system can be classified into vision-based and sensor-based. This work targets wearable-sensor HAR systems in healthcare, which are the most prevalent type of sensor-based HAR systems [**12**]. More importantly, wearable-sensor HAR systems do not suffer from severe privacy issues like vision-based HAR systems, making wearable-sensor HAR systems suitable for healthcare applications. In a wearable-sensor HAR system, a user wears portable mobile devices that have built-in sensors. The user's activities can then be classified by measuring and characterizing sensor signals when the user is conducting daily activities.

HAR for healthcare has many potential use cases, including (1) Moving gait diagnosis from expensive motion labs to the community. Gait analysis can be used in many healthcare applications, such as stroke detection, gait modification (to prevent failing), and certain disease early detection. (2) Cognitive behavior monitoring and intervention for children and adults with attention-deficit/hyperactivity disorder (ADHD). We can leverage sensors to investigate whether fidgeting positively or negatively affects attention. (3) Stroke-patient hospital direction. When a patient is in an ambulance, a life-and-death question is whether the patient has extensive brain hemorrhage. If so, the patient should be directed to a hospital that can treat such cases. UCSF has developed a device based on an accelerometer sensor to help make this critical decision. (4) Epilepsy and Parkinson's disease study. Doctors have collected a significant amount of data on electrophysiology and episodic memory in rodents and human patients. The analysis of such sensing data can be used for various disease identification and treatment purpose. (5) An expensive device, called Vision RT, is used to ensure radiation therapy is delivered safely to cancer patients (due to patient motion). It is worth exploiting sensors to detect the patient's movement while taking radiation therapy for the less affluent communities.

However, building practical wearable-sensor HAR systems for healthcare applications not only has challenges (e.g., sensor setup, data collection, and AI model selection) that are faced by traditional wearable-HAR systems, but also challenges that are unique to the healthcare domain. For example, in addition to the overall AI model accuracy (averaging results of all users), clinicians are concerned about the model stability (i.e., the model has approximately the same accuracy for each user) and model interpretability (e.g., to discover patient movement patterns that are specific to some symptoms).

Therefore, we present this work in the hope to shed light on designing wearable-sensor HAR systems for healthcare applications. To illustrate the system considerations, we share two of our healthcare systems: one for identifying the early mobility activities of ICU patients [**13**] and the other one for the gait analysis of DMD patients [**14**]. Our projects demonstrate that HAR systems for healthcare not only have commonalities such as data processing pipelines but also differences in terms of sensor setup and system requirements.

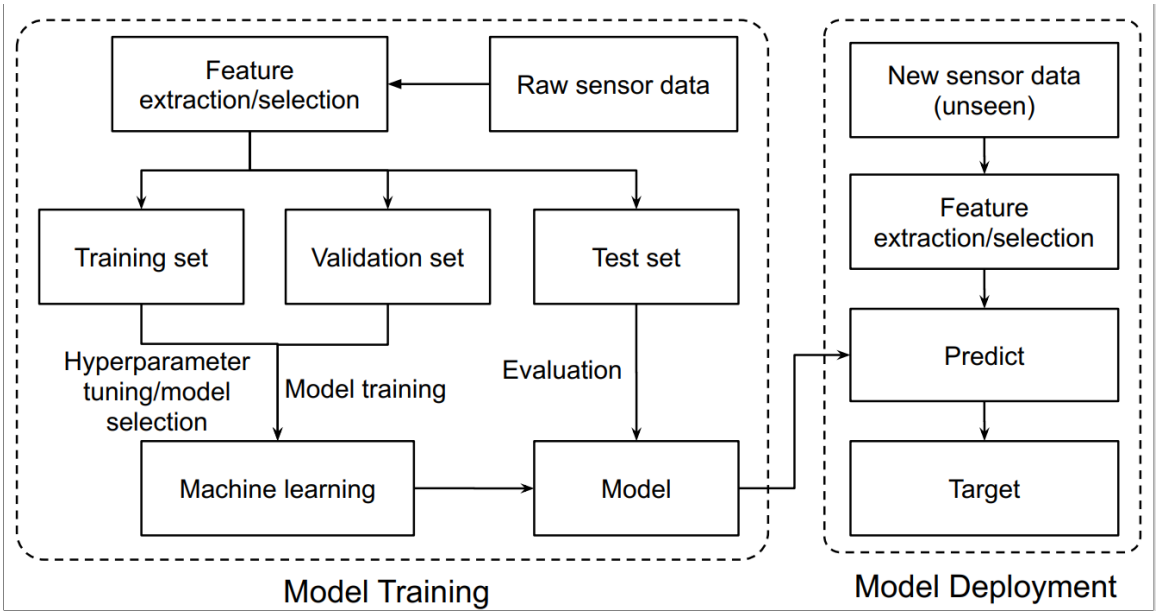


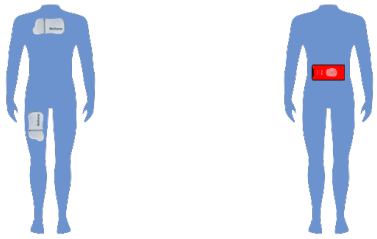
FIGURE 2.1. General data flow for the two-stages of HAR systems: model training and model deployment.

We organize this work as follows. First, we provide the preliminaries of HAR systems. Next, we introduce our HAR systems for ICU patients and DMD patients. Then, we explain the considerations when designing a HAR system. Last, we highlight the challenges of applying wearable-sensor-based HAR systems to healthcare, and propose several research opportunities. Last, we conclude this work.

2.2. Human Activity Recognition: A Primer

Given the short time-length data of wearable sensors, a HAR system needs to recognize the activity from which the data is generated. Thanks to the rapid advancement of AI technology, AI algorithms/models are increasingly adopted for recognizing the activity from the sensor data. Figure 2.1 illustrates the general data flow for an AI-based HAR system, which can be divided into two stages: model training and model deployment.

In the model training stage, an AI model is trained and tailored for the specific application. To achieve an accurate AI model, the following steps are often applied. First, raw sensor data from different activities should be collected. The quality of collected data significantly affects the AI model performance. The collected data is required to be diverse, representative, and large in the



(a) ICU-patient (b) DMD patient

FIGURE 2.2. Device setups in our HAR projects. (a) We use two accelerometer devices to recognize the early mobility activities of ICU patients. One device is on the chest and the other device is on the thigh. (b) We use one smartphone that captures accelerometer data to identify DMD patients. The phone is located at the backside body.

number of samples. Afterward, the raw data is divided into fixed-length or dynamic-length segments (i.e., time windows) [**15**]. Then, feature extraction is used to extract potentially useful features from the data segmentation, and feature selection is adopted to remove irrelevant features [**16**]. To alleviate the overfitting problem of the trained model, the set of processed features are divided into a training set, a validation set, and a test set. During the AI model training, we use the training set to tune the AI model and the validation set to measure the model's accuracy. After we finish the model training, we use the test set to evaluate the trained model. The trained model is deployed to real-world applications if its accuracy is satisfactory. Otherwise, the whole model training stage is performed repetitively by exploring different configurations, such as applying other feature extraction methods and changing AI models.

In the model deployment stage, the same data processing (e.g., segmentation, feature extraction, and selection) is applied to the new and unseen sensor data, and the trained model is executed on the processed data. It is possible that the trained model may not work as expected in a real deployment, probably due to the model over-fitting or the lack of generality in the collected dataset [**17**]. In this situation, the system designer needs to revert to the model training stage.

2.3. HAR Applications in Healthcare

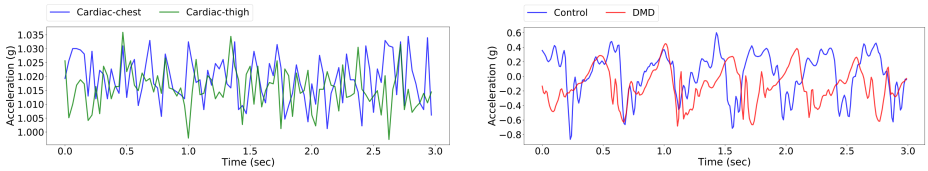
Clinicians have already applied wearable sensor-based HAR systems in healthcare, thanks to the development of more lightweight wearable devices, greater computation capability, and higher

accurate AI algorithms. This section presents our two HAR healthcare projects to illustrate the considerations when designing HAR systems for healthcare applications with different goals.

2.3.1. Case 1: Identification of Early Mobility Activity for ICU Patients. Due to long periods of inactivity and immobilization, patients become weak when recovering from major illnesses in ICU [**18**]. If ICU patients' activities can be accurately recognized, clinicians can provide an optimal personalized dose of mobilities for ICU patients' different illness conditions. Therefore, doctors and researchers are extremely interested in ICU patients' early mobilization, which is an effective and safe intervention to improve functional outcomes [**19**]. However, early mobility activity (EMA) research is limited by the lack of accurate, effective, and comprehensive methods to recognize patients' activities in ICU.

We propose a wearable sensor-based HAR system for recognizing the EMA of ICU patients [**13**]. In our system, Each ICU patient wears two accelerometer devices: one on the chest and the other on the thigh, as shown in Figure 2.2(a). Each device continuously collects 3-axis accelerometer data at a sampling rate of 32 Hz. Figure 2.3(a) plots the accelerometer data when an ICU patient sits on the cardiac chair to achieve an optimal resting position. This project aims to classify 20 types of ICU-related activities (e.g., reposition, percussion).

This project has two main challenges in designing the HAR system for ICU patients. (1) Label Noise. Because the time lengths for accomplishing an early mobility activity are different for ICU patients with varying health conditions, it is laborious and time-consuming work for clinicians to annotate sensor data for each second in the real world. Therefore, our EMA sensor data are annotated for each minute by a medical expert after data collection. However, one-minute length is exceedingly long for some early mobility activities such as Reposition, which the patient needs less than 20 seconds to accomplish. This annotation process introduces the label noise in our EMA dataset, which decreases the accuracy of the model. (2) Sensor Orientation. In the actual data collection process and possible future applications, we cannot guarantee that the orientations of all accelerometers are the same, and different orientations of the accelerometers lead to different meanings of XYZ coordinate values. Therefore, without careful feature extraction and selection, the AI model generalizes poorly to different patients, affecting the system performance in practice.



(a) ICU patient (b) DMD patient

FIGURE 2.3. Illustration of accelerometer data in our projects. (a) The z-axis of the accelerometer data from the two on-body devices when an ICU patient is performing the cardiac activity. (b) The z-axis of the accelerometer data, which shows the difference in gait characteristics between a DMD patient and a healthy person.

To tackle these challenges and improve the accuracy of recognizing ICU patient's activities, we explore the following techniques. (1) We propose a segment voting process to handle the label noise. Specifically, each one-minute sensor data is divided into multiple fixed half-overlapped sliding segments (time windows). We train our AI model using the segments. To predict each one-minute sensor data activity, we apply our trained model to each segment. The final prediction result for the one-minute data is the activity that has the majority vote among the prediction of all segments. Our segmenting method improves the model accuracy by ∼4.08% and reduces the model instability by ∼9.77% [**13**]. Our experiments also demonstrate that the number of sensors contributes to eliminating label noise in our dataset. As shown in Figure 2.3(a), the increase in the number of sensors conveys more information, and thus improves the system's accuracy. (2) We identify and extract features that are not sensitive to sensor orientations to tackle the sensor orientation problem. Our features improve both the accuracy and the stability of AI models compared to the model trained on commonly used features.

2.3.2. Case 2: Identification of Gait Characteristics for DMD Patients. Duchenne muscular dystrophy (DMD) is a genetic disorder disease that affects the dystrophin protein, essential for keeping muscle cells intact. It has an estimated incidence of 1:5000 male births, and untreated boys become wheelchair-bound by the age of 12 years and die in their late teens to early 20s [**20**]. There is presently no cure for DMD disease. Nonetheless, gene repair interventions and other preventive therapies can be initiated as early as possible to slow the disease's progress and prevent secondary conditions. Therefore, it is important to identify children with DMD early in the course

of their disease and have tools for quantitative evaluation of their gait in both the clinic and community environments.

We designed a wearable sensor-based HAR system to identify gait characteristics associated with the progression of gait abnormalities in children with DMD and to differentiate those patterns from those of typically developing peers [**14**] To leverage this idea, we design a HAR system in which we use a smartphone to capture accelerometer data from the participants. As Figure 2.2(b) illustrates, participants wear a smartphone at the back of the hips over the spine (lumbosacral junction) at a location that is the closest surface point to the body's center of mass. Each smartphone collects 3-axis accelerometer data at a sampling rate of 30 Hz with the same phone orientation.

We recruited ambulatory participants with DMD between 3 and 13 years of age and typically developing controls of similar ages. We ask participants to perform exercises at various times, speeds, and distances such as free walk and 6-minute walk, as specified by the north star ambulatory assessment (NSAA) standard [**21**]. Figure 2.3(b) shows the gait pattern difference between a DMD patient and a healthy person when they are walking.

We found that classical machine learning and deep learning, after hyper-parameter fine-tuning and cross-validation on seven different gait activities, led to the best performance with an accuracy exceeding 91% on the 6-min-walk-test activity [**14**]. We demonstrate that by using AI techniques and an accelerometer, we can distinguish between the DMD gait and typically developing peers.

There are two main challenges in designing our HAR system for the DMD application: clinical interpretability and data sparsity. (1) Clinical Interpretability. Medical practitioners desire not only a high prediction accuracy but also an interpretation of the prediction result. (2) Data Sparsity. In the healthcare domain, collecting diverse and sufficient data is challenging, especially for fatal diseases such as DMD.

We explore the following techniques to tackle these challenges. (1) To interpret AI model outcomes, we plan to link the clinical measurements with the model's extracted features by leveraging advanced AI models such as interpretable CNN [**22**]. However, it is an active, challenging task to find which clinical measurements correlated with the AI model features, especially for deep learning models. (2) To overcome the lack of data, we plan to use Generative Adversarial Network

(GAN) [**23**] or synthetic minority over-sampling technique (SMOTE) [**24**] to generate more data samples.

2.3.3. Summary of Our Projects. Our two projects target different healthcare applications with different goals: recognizing ICU patients' activities and distinguishing DMD gait patterns from those typically developing controls. The ICU project focuses on the system performance to assist the doctor in better understanding patients' recovery. While achieving high system performance, the DMD project interprets the model results further and discovers disease-specific patterns to determine the patient's condition and progression. Our example projects demonstrate the effectiveness and potential of wearable sensor-based HAR systems in healthcare. However, due to the different goals, different healthcare applications may have additional HAR system considerations. For example, our two projects adopt a different number of devices (2 versus 1) and device position (chest and thigh versus central mass body). In addition, our projects also apply different feature extractions (time and frequency domain versus clinical). In the next section, we present design considerations for building HAR systems.

2.4. System Design

This section covers three design considerations essential for HAR systems, i.e., sensor, feature extraction and selection, and AI model selection.

2.4.1. Sensor. Sensors play an essential role in wearable HAR systems. Different HAR systems adopt various sensor configurations regarding the type of sensors, the sensor position and orientation, and the number of sensors.

2.4.1.1. *Sensor Types.* There are several types of sensors. Each sensor captures a different raw movement signal. The most commonly-used wearable sensors in HAR systems are accelerometer, gyroscope, and electrocardiography (ECG). The accelerometer sensor captures the acceleration signal that is useful for recognizing movements such as walking, running, and jumping. Gyroscopes capture the rotation movements used commonly in recognizing swinging, turning, and repositioning. ECG captures the heart rate and rhythm, which helps distinguish between intensive and light exercises.

However, many activities include both directional and rotational movements. Therefore, using one sensor type is not adequate. As a result, multiple types of sensors (e.g., accelerometer and gyroscope) are used in various application scenarios to maximize accuracy. However, using multiple types of sensors is challenging due to the increased complexity of the system in terms of synchronization issues [**6**].

2.4.1.2. *Sensor Position and Orientation.* Different positions and orientations of devices affect the data features and thus the model accuracy in predicting different activities [**25**]. However, there have not yet been systematic comparisons of the number, type, and location of sensors to determine whether an optimal array design can capture data across a wide range of human activities and disease states. In many cases, the device position and orientation are decided by the empirical experience of clinicians.

2.4.1.3. *Number of Sensors.* Generally, a large number of sensors require demanding storage and computation capability. On the other hand, more sensors can collect more diverse data, which is beneficial for improving model performance [**26**]. Therefore, to decide the optimal number of sensors, researchers need to carefully consider many factors such as cost, power consumption, and accuracy target as well as the feasibility of long-term use in the community to collect real-world information [**27**].

2.4.2. Feature Extraction and Selection. In addition to the hardware setup, feature extraction and selection significantly affect the overall system performance. Before applying feature engineering to the data, the input data needs to be segmented.

2.4.2.1. *Data Segmentation.* HAR systems collect data constantly via wearable sensors to identify possible activities. Data segmentation is applied to divide comparatively long time data into short fragments (time windows) that are suitable for AI models to learn. There are two types of data segmentation: fixed-length and dynamic-length [**15**]. For fixed-length segmentation, if the time window is too short, the extracted features from the fragments are insufficient to capture the activity; on the other hand, if the time window is too long, a fragment is likely to contain multiple activities. The system accuracy deteriorates in both cases. In comparison, a dynamic-length data segmentation adopts an adaptive length of fragments corresponding to the characteristics of input

data. Ideally, dynamic data segmentation generates fragments, in which each fragment only contains a single and complete activity. However, dynamic data segmentation is much more complex than fixed data segmentation, and thus are not as widely adopted by existing works as fixed-length segmentation.

2.4.2.2. *Feature Extraction.* Feature extraction is then applied to extract important features from the data fragments [**16**]. It can be broadly classified into time-domain and frequency-domain methods. In time-domain feature extraction, metrics such as median, variance, mean, and skewness are calculated over the amplitude variations of data over time. Time-domain features are lightweight to compute and thus are friendly to low-profile embedded devices and real-time applications. In comparison, frequency-domain features calculate the frequency variations of data over time. They include metrics such as spectral entropy, spectral power, and peak frequency. The computation overhead of frequency-domain features is generally much greater than time-domain features. In reality, most existing HAR systems adopt both time-domain features and frequency-domain features, in the consideration of the tradeoff among factors such as system accuracy, computation overhead, and power consumption.

2.4.2.3. *Feature Selection.* Feature selection is often adopted in order to reduce system complexity. It measures the importance of features and then removes irrelevant features. Feature selection is roughly divided into three methods: filter methods, wrapper methods, and embedded/hybrid methods [**16**]. Filter methods select a subset of features by exploiting inherent characteristics of features, whereas wrapper methods use classifiers to estimate the useful features. On the other hand, the embedded/hybrid methods combine the results from filter methods and wrapper methods [**12**]. By carefully selecting features, the AI model accuracy can be significantly improved. However, in healthcare HAR systems, pursuing high accuracy is not the sole goal, as the features are often manually decided by medical experts for identifying patients. Therefore, healthcare HAR systems require feature extraction and selection that is meaningful for clinicians and meanwhile achieves high prediction accuracy.

2.4.3. AI Model Selection. In the HAR field, classical machine learning algorithms and deep learning algorithms have been explored and applied, which is summarized in Figure 2.4. Both

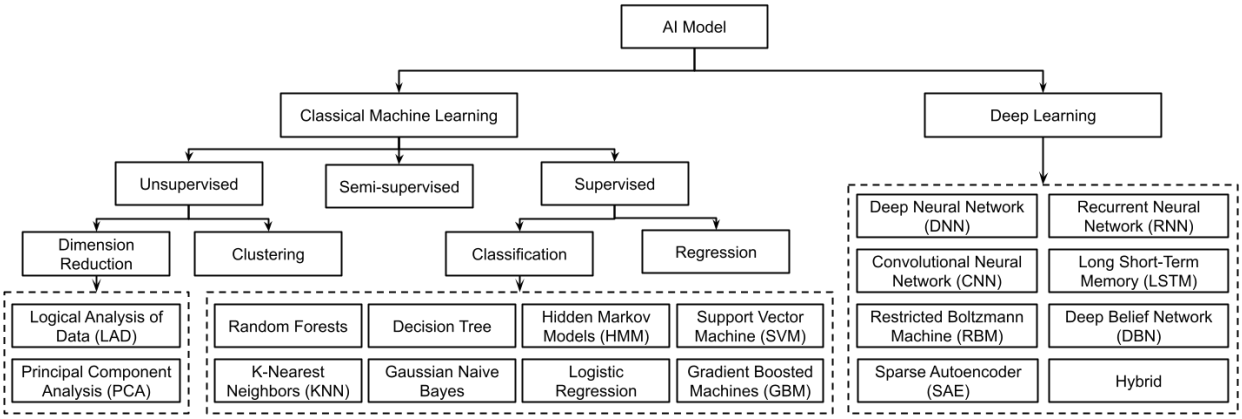


FIGURE 2.4. Classical machine learning and deep learning algorithms used in HAR systems.

classical machine learning algorithms and deep learning algorithms have different advantages and disadvantages.

Dataset requirement and system running overhead. The data collection process in the healthcare scenario is challenging because of the severe privacy issue and rare incidence rate of some medical activities. Therefore, in most healthcare applications, the database size is small. Correspondingly, classical machine learning models are more preferred because they work well with medium-size datasets. In contrast, even though deep learning models achieve better accuracy, they usually require a large amount of data for training. Real-time performance is another critical factor for some healthcare applications [**28**]. For example, [**29**] uses cranial accelerometers to detect stroke in an ambulance to decide whether to send the patient to a specialist stroke hospital for special treatment. Therefore, lightweight models are preferred in this use case. In addition to the running overhead of the AI models, the processing time of feature extraction also affects the model selection, because different model structures adapt differently to the extracted features.

System interpretability. The features extracted from the sensor data are helpful to understand the pattern of some specific diseases to find out the pathological characteristics of the disease. For example, we extract the temporal/spatial gait characteristics from sensor data to evaluate the gait changes associated with DMD. Classical machine learning models are easier to interpret the model's decision, especially in decision tree models. Even though there is a great deal

of work in interpreting deep learning models, deep learning models have the reputation of poor interpretability.

2.5. Challenges and Opportunities

Wearable sensor-based HAR systems are promising for a variety of healthcare problems. However, there are several challenges in fully exploiting them to build satisfactory HAR systems for healthcare. In this section, we identify challenges as well as research opportunities of HAR systems for healthcare.

2.5.1. Data Sparsity. The most commonly used algorithms for the HAR system in healthcare are the supervised learning algorithms that need extensive labeled data. For some daily living activities such as walking and running, researchers could get a significant amount of the labeled data from the public dataset or the raw sensor data collected and annotated by themselves. However, for some specific human activities related to healthcare, such as the therapeutic activities of patients, researchers could not get enough sensor data since these activities are low-probability events compared with daily life activities. Furthermore, it also takes time and effort to locate the sensor data of these specific activities from the daily data and label them. For example, when patients recover from surgery, they need some range of motion(ROM) exercises several times a day to make their joints and muscles flexible and strong again. Because of the fixed and limited collection times per day and the limited number of patients are involved, raw sensor data for ROM becomes insufficient, affecting the HAR system's performance. Therefore, building HAR systems with high accuracy on small datasets in healthcare is one of the most significant challenges.

Meta-learning is one of the approaches to solve this challenge. Meta-learning aims to optimize models which can learn efficiently in a small dataset when dealing with new categories. In [**30**], researchers present a meta-learning methodology based on the Model-Agnostic Meta-Learning algorithm [**31**] to build personal HAR models. In [**32**], researchers use few-shot learning to transfer information from existing activity recognition models. However, it is unclear whether these techniques work well for medical applications. So more research is needed to explore the feasibility of transferring knowledge from daily living activities to specific activities related to healthcare.

2.5.2. Model Interpretability. In HAR applications in healthcare, an increasing number of applications focus on the interpretability of the model to extract relevant features, in order to describe the severity of the disease and track the progression of the disease [**14**]. In addition, notwithstanding the benefit of deep learning in HAR, the underlying mechanics of machine learning are still unclear. So, various studies are trying to explain the deep learning model for the recognition of human activities. The common approach to interpreting the deep learning model is to compute the importance of each part of the input. In [**33**], researchers propose an interpretable convolutional neural network to select the most important sensor position for some specific activities. Instead of computing the importance of each part of the input, another approach is to make a sequence of selections about which part of the input is essential for the model training [**34**]. More research is required to adopt these methods to HAR systems for healthcare.

2.5.3. Concurrent Activities. Most of the existing HAR research focuses on single-labeled activity, recognizing only one activity of the given data segment. However, in real-world healthcare scenarios, humans can perform multiple activities concurrently. For example, patients can do ROM exercises and percussion therapy at the same time. The AI model performance deteriorates for concurrent activities. On the other hand, designing models to recognize multiple activities per data segment is a challenging task.

2.5.4. Composite Activities. In healthcare applications, optimizing HAR algorithms to identify composite activities in the community is ultimately more desirable than recognizing a single type of task. For example, when a patient moves from bed to the chair, the patient performs various activities, including sitting from supine in the bed, pivoting to place feet on the floor, standing from sitting, walking a few steps, and then sitting down on a chair. Therefore, it is preferred that an AI model can directly recognize the composite activity.

2.5.5. Privacy. Wearable sensor-based HAR systems do not suffer from severe privacy issues as camera-based vision systems. However, since HAR applications continuously capture user data and recognize user activities, they may leak users' personal information if data are not secured. Therefore, secure data sharing and safe data storage are imperative for healthcare applications. To alleviate sensitive information during model training, adversarial loss functions are leveraged to

guard against privacy leakage [**35**]. In addition, federated learning is a promising solution, which trains a global model without exposing local devices' private data [**36**].

2.5.6. Opportunities of HAR for Healthcare. Through our experience with HAR systems for healthcare, we identify the following research opportunities.

- **Community-based healthcare**. Community-based healthcare requires that user devices are lightweight and affordable for the public. In addition, instructing the non-expert users/patients should be straightforward to follow. We can use digital sensing capability and the popularity of mobile devices to enable large community-based prescreening for various diseases and early signs of diseases. This can be done in a privacy-preserving manner in the sense that data does not need to leave a local device if necessary. For example, our DMD project enables community-based diagnosis during the pandemic and in rural areas where specialty labs are hundreds of miles away.
- **Chronic disease prevention and intervention.** For chronic diseases, it is essential to capture the behaviors of patients in the long run. To this end, gait analysis, motion monitoring, ECG, and other vital signals (such as continuous glucose monitoring) can play a key role.
- **Health aging**. With the decreased fertility rates and the increased life expectancy, population aging is becoming common for most countries. Therefore, building HAR systems for healthy aging is beneficial for senior citizens and society as a whole. We anticipate that gait and motion monitoring and diagnosis will play a critical role in healthy aging.

2.6. Conclusion

It is gaining popularity by applying wearable sensors to recognize and analyze human activities for the healthcare domain. For example, we leverage HAR systems to recognizing patients' early mobility activities in ICU and to analyzing the symptoms of DMD patients. This overview work covers the system design of HAR systems based on wearable sensors, focusing on healthcare applications. We emphasize the essential components of HAR systems, including sensor factors, data segmentation, feature extraction and selection, and AI model comparison. We also highlight the challenges and opportunities of HAR systems for healthcare.

CHAPTER 3

Early Mobility Activity Recognition for Intensive Care Unit Patients Using Accelerometers

With the development of the Internet of Things(IoT) and Artificial Intelligence(AI) technologies, human activity recognition has enabled various applications, such as smart homes and assisted living. In this work, we target a new healthcare application of human activity recognition, early mobility recognition for Intensive Care Unit(ICU) patients. Early mobility is essential for ICU patients who suffer from long periods of bedrest. We present an AI model to recognize patients' early mobility. To improve the model accuracy and stability, we identify features that are insensitive to sensor orientations and propose a segment voting process that leverages a majority voting strategy to recognize each segment's activity. Our results show that our algorithm improves model accuracy from 77.78% to 81.86% and reduces the model instability (standard deviation) from 16.69% to 6.92%, compared to the same AI model without our feature engineering and segment voting process.

3.1. introduction

Due to long periods of inactivity and immobilization, Intensive Care Unit(ICU) patients become weak when recovering from critical illnesses [**37**]. Early Mobility(EM) is an effective and safe intervention to improve ICU patients' outcomes, such as ventilator days and functional status [**38**]. Therefore, it is of great interest for clinicians to identify ICU patients' early mobilization. When clinicians can accurately recognize the EM activities of the ICU patients, they can prescribe an optimal personalized dose of mobility to the ICU patients. However, the advancement of research on EM is limited due to the lack of accurate and effective systems to quantify patients' EM activities in the ICU [**39**].

EM recognition is a sub-topic of Human Activity Recognition, a fast-moving area because of the recent advancement of Internet of Things(IoT) and Artificial Intelligence(AI) technologies. Specifically, IoT technology enables convenient data acquisition and edge computing capacity, while AI technology provides accurate and efficient machine learning algorithms. Consequently, HAR systems have been applied in various fields such as fitness, smart homes, and assisted living [**6**]. HAR healthcare applications facilitate disease detection and provide proactive assistance to quantify movement for both patients and clinicians, etc. For example, Goschenhofer et al. built a HAR system to analyze the symptoms of Parkinson patients [**40**]; Palaniappan et al. built a proactive assistance system for senior citizens by identifying their abnormal activities [**41**].

Based on the type of data being collected, a HAR system can be classified into vision-based [**28**], and motion sensor-based [**42**]. A vision-based HAR system usually deploys cameras at fixed locations of interest and applies computer vision techniques to recognize human activities. However, such systems have severe privacy issues, especially in the ICU. In comparison, a sensor-based HAR system leverages lightweight sensors that patients wear. Different types of sensors can capture different features of activities, e.g., movements by accelerometer and gyroscope [**43**], environment by temperature and humidity sensors [**44**], and physiological signals (e.g., heart rate) by electrocardiogram sensors [**45**]. We leverage accelerometers to recognize ICU patients' EM activities because accelerometers are lightweight, energy-efficient, and widely available.

In our system, each patient wears two accelerometer sensors: one sensor is placed on the chest, and the other sensor on the thigh. Patients perform EM and routine activities in the ICU room while the sensor data is continuously collected. Our system automatically recognizes patients' EM activities based on the sensor data. Since our system targets applications in ICU, it needs to have both high **accuracy** and **stability**. High accuracy means that the system achieves satisfactory average recognition precision of all patients, while high stability emphasizes that the recognition accuracy for different patients is approximately the same.

There are three main challenges in realizing our system. (1) *Distorted Activities.* EM activities are different from traditional human activities in the HAR system. Traditional human activities usually consist of regular activities with periodic patterns, such as running, walking, and swimming [**46**]. In comparison, ICU patients' EM activities are distorted, take much longer than regular

activities, and vary among patients. Also, they may be affected by medical equipment, and physical therapy personnel, who assist patients with exercises, such as Range Of Motion(ROM), which is an EM activity that helps patients regain joint or muscle strength. (2) *Label Noise.* A medical expert annotates the activity of each minute's sensor data. However, for ICU patients with different health conditions, the time needed to accomplish an activity is significantly different. Furthermore, intermittent behaviors during the activities are observed. For example, one patient could finish an EM activity once in a minute while another patient could accomplish the same EM activity twice in a minute or accomplish this same EM activity and other non-EM activity in a minute. Hence, the label for every minute's sensor data is coarse, leading to the label noise. Label noise is an important issue for classification tasks, which decreases the accuracy and the stability of AI models [**47**]. (3) *Sensor Orientation.* Although sensors are placed at roughly the exact locations (thigh and chest) on our ICU patients, their orientations are different. As a result, the 3-axis sensor readings (X, Y, Z-axis) have different physical meanings for different patients. Therefore, without careful feature extraction and selection, the AI model generalizes poorly to different patients.

To the best of our knowledge, our work is one of the first HAR systems for recognizing ICU patients activities using wearable sensors. In realizing our system, we make the following contributions: (1) A system for EM recognition for ICU patients. Our system adopts two accelerometer sensors in different positions to capture different movement features of EM activities. Through a combination of sensors on the chest and on the thigh, the system collects more information to identify distorted movements. (2) We propose a segment voting process to handle the label noise problem. Specifically, each one-minute sensor data is divided into multiple fixed half-overlapped sliding segments (time windows). We train our AI model using the segments. To predict the activity of each one-minute sensor data, we apply our trained model to each segment. The final prediction result for the one-minute data is the activity that has the majority vote among the predictions of all segments. (3) To tackle the sensor orientation problem, we identify and extract features that are not sensitive to sensor orientations. Our features improve both the accuracy and the stability of AI models compared to the model trained on commonly-used features.

We evaluate the accuracy and the stability of our system for classifying two categories of lying activities for ICU patients. To objectively evaluate our system for new patients, we adopt

leave-one-patient-out cross-validation. The experimental results show that our system increases the classification accuracy from 77.78% to 81.86% and reduces the model instability (standard deviation) from 16.69% to 6.92%, compared to the model without our feature engineering and segment voting process.

3.2. Related Work

This section presents related work on machine learning algorithms for HAR, HAR applications in the healthcare domain, and the community's endeavor to assist ICU patients.

Machine Learning Algorithms for HAR. Different machine learning algorithms (classical machine learning and deep learning) are adopted for recognizing activities. For example, classical machine learning algorithms such as Decision Tree [**48**] and Logistic Regression [**49**] are applied to the extracted clinical features. Deep learning algorithms such as convolution neural networks are recently leveraged, which achieve higher accuracy [**50**]. We focus on classical machine learning algorithms because 1) our collected dataset is too small for deep learning (only 586 data samples), and 2) doctors prefer feature-based algorithms in order to interpret the results.

HAR for Healthcare Applications. Different medical applications face different challenges and considerations, so various HAR systems are designed to target different applications. For example, Patel et al. proposed a Support Vector Machine (SVM)-based approach to evaluate the severity of symptoms for patients with Parkinson's disease based on wearable sensor data [**51**]; Kańtoch proposed a neural network classifier to assist the recovering of patients by recognizing squats (a common rehabilitation exercise) [**52**]; Jennifer et al. targeted aging populations in the home to assist living such as abnormal activities detection and security assurance [**53**]. As far as we know, our system is one of the first works on HAR, specifically tuned for ICU patients using wearable sensors.

HAR for ICU Patients. There are several works that leverage cameras to assist ICU Patients. For example, Yeung et al. developed computer vision algorithms to detect patient mobilization activities in the ICU, which achieves a mean area under the curve of 0.938 for identifying four types of activities [**54**]; Rishab et al. used depth cameras to mitigate the severe privacy issue of

(a) Sensor position (b) Accelerometer sensor

FIGURE 3.1. The sensor setup in our EM recognition system. (a) Two accelerometer sensors are worn by ICU patients. One sensor is on the chest, and the other sensor is on the thigh. (b) The clinical-grade 3-axis accelerometer sensor that is used in our system.

RGB cameras, and identify two types of EM activities [**55**]. In comparison, we use accelerometer sensors due to more widespread accessibility and also to avoid data loss when patients move out of camera range.

3.3. Data Collection

EM activities data were collected in individual patient rooms in a Medical ICU (MICU) of an academic medical center in California. Seventeen adult MICU patients who are eligible for early mobility interventions between 2016-2017 were recruited for data collection [**39**]. As Figure 3.1 illustrates, two 30Hz 3-axis accelerometer sensors are placed on each participant: one sensor is attached to the chest, and the other sensor is attached to the thigh(BioStampRC, MC10). Generally, continuous accelerometer data were collected over a period of 4 hours and up to 48 hours to ensure representative sampling of day and night mobility interventions and patient activities. Concurrently, all ICU activities were recorded using a camera system without audio. An ICU clinician reviewed and annotated each minute's sensor data activity based on the corresponding video. However, due to device and timing reasons, video or sensor data were missing for nine patients. In this analysis, eight patients with 586 minutes of sensor data for lying activities are annotated.

We classify two categories of lying activities as suggested by the American Association of Critical-Care Nurses (AACN) and the Society of Critical Care Medicine (SCCM), who define the

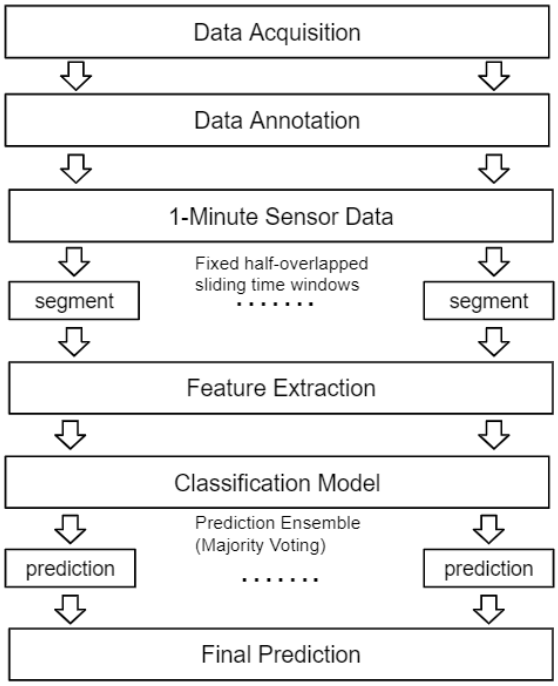


FIGURE 3.2. Data flow of our EM recognition system.

quantification approach of ICU early mobility [**56**]. Specifically, we focus on lying with no movement and lying with EM and ICU specific activities. The lying with EM and ICU specific activities includes four activities when patients lie on the bed; that is, repositioning, ROM exercises, percussion therapy, and oral care. Repositioning involves lying flat in bed and moving laterally from left to right side or vice versa; ROM exercises help recover patients' muscles and joints; percussion therapy assists patients with removing respiratory fluid tapped in the patients' chest for 8-10 minutes by clinicians; as well as, oral care are part of the daily activities performed in bed for ICU patients.

3.4. System Architecture & EM Classification

In this section, we first provide an overview of our system. Then, we explain our segment voting process and the features used in our system. Last, we introduce our machine learning model for classifying lying ICU activities.

3.4.1. System Overview. Figure 3.2 shows the data flow of our EM recognition system. Our system takes 1-minute sensor data from the two accelerometers as input. The data from these

two sensors are synchronized and integrated. Afterward, each 1-minute sensor data is divided into multiple half-overlapped sliding segments (time windows). Then, we extract time-domain and frequency-domain features that are insensitive to sensor orientations. To mitigate the noisy label problem of EM activities, we leverage the segment voting process, which predicts the activity of the 1-minute input as the activity that has the majority vote of segments' prediction results.

3.4.2. Extracted Features from Sensor Data. Both time-domain features and frequency-domain features are widely used in HAR systems [**42**, **57**, **58**]. Time-domain features reflect the trend of sensor signal changing with time, which can help the EM recognition system to capture the changes in activities [**59**]. In contrast, frequency-domain features capture dynamic motion in an activity, which can improve the EM recognition system to get the movement trend in an activity [**60**]. We use Fast Fourier Transform (FFT) to convert the sensor data from the time domain to the frequency domain to extract frequency-domain features. We extract the low-frequency components whose frequencies are less than 0.3Hz related to gravity's influence and the high-frequency components with frequencies between 0.3Hz and 20Hz related to the dynamic motion. And we also extract the derivation of each high-frequency component as our third frequency-domain features. Because different patients have different and unknown sensor orientations, we cannot rely on features that are sensitive to sensor orientations [**61**], such as features related to each axis. Instead, we apply the magnitude function to each tri-axial signal: original tri-axial signals, low-frequency components, high-frequency components, and the derivation of each high-frequency component. Finally, for each sensor position, we have one time-domain feature, namely the signal magnitude, and three frequency-domain features, namely the magnitude of low-frequency components, high-frequency components, and the derivation of each high-frequency component ,which are less susceptible to sensor orientation. Furthermore, we apply eight metrics (i.e., mean, maximum, minimum, standard deviation, median, and entropy) to each time and frequency-domain feature, resulting in a total of 64 attributes for training machine learning models.

3.4.3. Classification Model. Different machine learning algorithms have various advantages and disadvantages. Classical machine learning algorithms are more straightforward to interpret

based on the importance of features, which is especially helpful in the healthcare field. In comparison, deep learning algorithms do not require feature engineering and tend to have higher accuracy. However, the amount of training data and the training overhead of a deep learning model can be enormous. Our collected EM activities dataset contains 586 minutes of sensor data, and thus the number of samples is only 586 because each sample is 1-minute long. Therefore, deep learning models are not applicable to our project because of the limited data samples. Instead, we use classical machine learning models such as Logistic Regression, Bagging Decision Tree, and Support Vector Machine. Our experiments show that Bagging Decision Tree (Bagging for short) works best for our needs. Bagging is an ensemble meta-algorithm that reduces the high variance of the decision tree, and thus improves the accuracy and stability of the decision tree algorithm.

3.4.4. Segment Voting Process. The classification model takes each 1-minute data segment as input. Initially, the system accuracy is low because of the noisy label problem described earlier on in this work. To improve the recognition accuracy and model stability, we propose a segment voting process. In our segment voting process, 1-minute data is segmented into small pieces (segments) with a fixed half-overlap time window size. There exists a trade-off in selecting window size. When the time window size is short, each window is more likely to contain only one activity. However, the extracted features may be insufficient to distinguish different activities. On the other hand, when the time window size is long, each segment may contain multiple activities, and thus the recognition accuracy is reduced. Therefore, deciding an optimal time window size is crucial. Therefore, in our segment voting process, we explore different time-window sizes to find the optimal time window size for the EM activities dataset. After that, 1-minute sensor data is divided into data segments. Then we extract features from each segment and apply the classification model to each segment, which predicts the segment's activity. The activity of the 1-minute input is determined by selecting the activity that occurs most among the segments (i.e., majority voting).

3.5. Evaluation

In this section, we extensively evaluate our system concerning the model accuracy and stability. In addition, we explore how the time window size, the sensor positions and numbers, the extracted features, and the number of patients in the training dataset affect the system performance.

3.5.1. Implementation. All experiments are implemented using scikit-learn and Keras package, running on a server with Nvidia Titan Xp GPU and Intel Xeon W-2155. To objectively evaluate our system for new patients, we adopt leave-one-patient-out cross-validation. The model **accuracy** is calculated by averaging the prediction accuracy for all patients, while the model **instability** is the standard deviation of the prediction accuracy among all patients.

TABLE 3.1. Classification accuracy and stability when different time window sizes are used for segments.

	Baseline	Segment time window length			
	1-minute	4 sec	10 sec	20 sec	30 sec
Accuracy	77.78%	79.85%	**81.86%**	77.85%	78.9%
Instability	16.69%	7.89%	**6.92%**	13.01%	9.41%

3.5.2. Segment Voting Process. We propose a segment voting process to mitigate the noisy label problem. Our segment voting process divides each 1-minute sensor data into segments of smaller time windows. We evaluate the model performance when different time window sizes are used. We compare our performance with the baseline case that takes the whole 1-minute data as input.

Table 3.1 tabulates our system performance and the baseline performance. We have the following observations: (1) Our segment voting process improves both the system accuracy and stability. This is because our segment voting process mitigates the effect of noisy labels. By adopting a 10-seconds time window, we increase the system accuracy from 77.78% to 81.86% and reduce the system instability from 16.69% to 6.92%. (2) The time window length affects the system performance. When the time window size is too small, such as 4 seconds, the segment data is insufficient for the model to classify EM activities accurately. On the other hand, when the time window size is too large, such as 30 seconds, the accuracy also decreases because of overlapped activities in the segment. We find that a 10-seconds time window works best for our EM recognition. In the rest of the experiments, we set the time window size to 10 seconds.

Figure 3.3 plots the prediction accuracy for each patient with our segment voting process and compares it with the baseline. It clearly shows that our segment voting process results in better system stability than the baseline. Specifically, the baseline method has low prediction accuracy for patient ID 6 and 7, while our segment voting process still maintains good accuracy for both

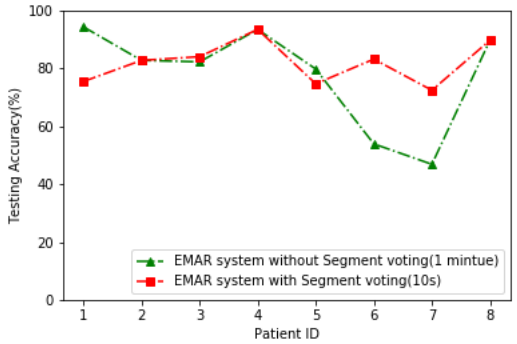


FIGURE 3.3. Our segment voting process has better system stability. In other words, we achieve similar prediction accuracy for different patients.

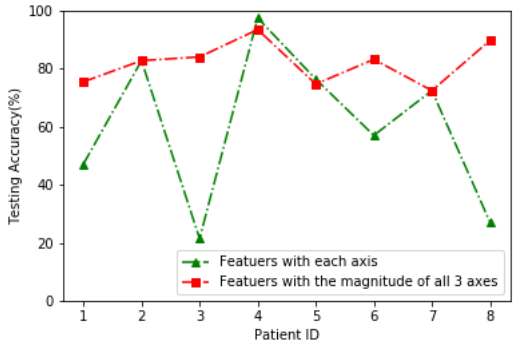


FIGURE 3.4. We extract features that are insensitive to sensor orientation, which improves accuracy and stability more than using commonly-used features.

patients. System stability is essential for healthcare applications, which require consistent diagnosis results for patients.

TABLE 3.2. The system performance for different sensor positions and sensor numbers.

Sensor	Chest only	Thigh only	Both
Accuracy	70.28%	80.29%	**81.86%**
Instability	16.98%	9.51%	**6.92%**

3.5.3. Extracted Features. We investigate whether our extracted features perform better than commonly-used features. Precisely, we extract features that are not sensitive to sensor orientations, while existing systems often include more features such as features for each axis' sensor data.

Figure 3.4 shows the performance of our extracted features which are only related to the magnitude of each tri-axial signal and commonly-used features, which contains the features of each axis. As we can see, our system has higher accuracy (especially for patients 1, 3, 6, and 8) and stability than the system with commonly-used features. Although the machine learning models can identify and extract the critical features, the limited dataset of ICU patients still requires careful feature engineering for the machine learning models to learn.

3.5.4. Sensor Positions and Numbers. In our current system, we use two sensors (one on the chest and the other on the thigh) to recognize the EM activities of ICU patients. We explore whether our sensor setting is superior to using a single sensor. For comparison, we adopt the same machine learning model but use one sensor signal as input.

Table 3.2 tabulates the system accuracy and stability when using only one sensor on the chest, only one sensor on the thigh, and using two sensors. The results clearly show that our sensor setting (both sensors) achieves better accuracy and stability than the single sensor counterparts. Specifically, the thigh-only scenario is better than the chest-only scenario in recognizing lying activities. Meanwhile, by combining both sensor data, our scenario further increases accuracy by 1.57% and stability (standard deviation) 2.59% compared to the best case of a single sensor (i.e., thigh-only scenario).

TABLE 3.3. The system performance when different numbers of patients are used in the training and testing.

#Patients	5	6	7	8
Accuracy	75.08%	78.35%	80.90%	81.86%
Instability	14.74%	11.99%	8.24%	6.92%

3.5.5. Number of Patients in Dataset. Our current system achieves 81.86% accuracy and 6.92% instability. Our system will improve if we have a larger dataset. To validate this hypothesis, we change the number of patients in our dataset and explore the system performance.

Table 3.3 tabulates the system performance when the dataset size varies from 5 to 8. The results shows that by increasing the number of patients, our system performance improves. It also suggests that our dataset is too small since the system performance has not yet reached the plateau. With a larger dataset, we expect that our system will improve further.

3.6. Discussion

In this section, we discuss some limitations of our work and present future works.

Our system adopts two accelerometer sensors and deploys them on the chest and thigh, which shows better accuracy and stability than the single sensor scenario. We believe that adding more sensors will improve the system performance at the expense of inconvenience for patients. The optimal number of sensors and sensor locations remains an open problem, especially for ICU patients with dramatically different activity patterns.

Our system achieves 81.86% accuracy, which may seem moderate considering the much higher accuracy reported by existing HAR papers. However, we target challenging activities of ICU patients in a real-life setting, while existing works mainly focus on activities performed by healthy people who are consistent in conducting activities such as walking and running, often in a controlled and protocolized setting. Furthermore, we use leave-one-patient-out cross-validation, which is the patient-level validation; i.e., we hold out all data from a patient and use it for evaluation. This validation process is a more robust approach for recognizing human activities than observation-level validation which we split the dataset into 85% for the training dataset and 15% for the testing dataset. Our results using observation-level validation show a much higher accuracy which is 97.9% by using Convolutional Neural Networks with segment voting process. Therefore, since observation-level validation may lead to a significant bias on unseen patients, patient-level validation is practical as a trained model in healthcare. In addition, from Table 3.3 we can see that our system has not fulfilled its full potential because the data sample size is still small. Our conjecture is that by collecting more data samples, our system's accuracy may increase further.

In the future, we plan to consider more activities in addition to lying activities. As we have shown in this work, recognizing EM activities of ICU patients involves many challenges, even for distinguishing only two categories of activities. Therefore, more research is required to support EM recognition for ICU patients using wearable sensors entirely.

3.7. Conclusion

This work presents novel work for EM recognition and clinical interventions of ICU patients using wearable sensors. Due to ICU patients' irregular movement behaviors, EM recognition for ICU

patients is more challenging than traditional human activity recognition (e.g., walking and running). By systematically considering the sensor positions and numbers, machine learning models, extracted features, time windows, and segment voting process, our system achieves good accuracy (81.86%) and stability (standard deviation of 6.92%) for discriminating between two categories of lying activities specific to ICU patients. More research is required to support higher system performance and more activities of ICU patients.

CHAPTER 4

BWCNN: Blink to Word, a Real-Time Convolutional Neural Network Approach

Amyotrophic lateral sclerosis or ALS is a progressive neurodegenerative disease that affects nerve cells in the brain and the spinal cord. While an individual affected by this disease loses all motor functions, they can still blink their eyes. We present a system (BWCNN) to use the eye blinks as a mode of communication with the outside world. The system uses a Convolutional Neural Network (CNN) to predict the state of the eyes and this state is used to find the blinking pattern. Once the pattern is obtained, it is mapped with a collection of phrases with a label for each pattern. Several state-of-the-art ConvNet architectures such as ResNet, SqueezeNet, DenseNet, and InceptionV3 were implemented in this system to evaluate their performance in order to choose the best one for eye state recognition. By tuning the hyperparameters of the networks such as batch size and the number of iterations, we analyzed the performance of different architectures based on the accuracy and the time taken for prediction. The goal of our system is to strike a balance between accuracy and the latency. High accuracy makes our prediction better while small latency makes it faster. Although the highest accuracy was obtained from ResNet (99.26%, 117ms latency). We found that InceptionV3 architecture struck the best balance for our system with an accuracy of 99.20%, and 94ms latency.

4.1. Introduction

There are several medical disorders that cause an individual to be paralyzed. In most cases, the patient loses the ability to communicate with the outside world even though their intelligence is still not affected. The specific problem here is to create a communication mechanism to aid paralyzed individuals to converse by blinking the eyes. The system needs to be robust and provide accurate results in real-time.

We have created a vision-based system to tackle all the limitations of the previous methods. Our system uses an InceptionV3 architecture and we have achieved an accuracy of 99.20% compared with around 96% accuracy in [**62**]. It is a completely safe method unlike the method based on Infrared [**63**] which causes cataracts. (Our system detects the state of eyes 'Open' and 'Closed' even under low lighting conditions, unlike the Haar-cascade based methods [**64**]).

We have a pre-defined set of patient inputs corresponding to the blink pattern which we map to actions in real-time. The inputs are how the patient interacts with our visual interface. These inputs could correspond of to movement (up, down, left, right), or clicks, etc. which would enable them to use different applications such as a browser, email, or phrase keyboard. As a proof of concept, we mapped those inputs to specific sentences. Since it uses predefined phrases instead of using Morse code or any other encoding patterns, the patient does not have to put in a lot of effort to spell out the entire sentence.

4.2. System model

In order to sum up, we want to create a system that works almost flawlessly in real-time and is completely safe to use and this can be summed up using (4.1) and (4.2). P refers to the performance of the system in our case is the accuracy. S refers to the System itself. W refers to the weights that are going to be trained to achieve this performance. A refers to the architecture that is going to maximise the performance. AS refers to the set of all architectures that can be used for this purpose. tuning time is the prediction time for the model on the validation set and TC is the time constraint which is 100 ms for our case.

$$\max P(S, W, A); A \in AS \tag{4.1}$$

$$s.t\ prediction\ time \leq TC \tag{4.2}$$

4.2.1. Previous work. Certain approaches have been used in the past to solve this problem. One approach [**65**] is to use Infrared (IR) sensors to estimate the state of the eyes to detect blinking and conversion to Morse code. This approach had multiple shortcomings such as the IR sensor

getting irradiated by other sources resulting in false eye-blinks and prolong usage may involve an increased risk of cataract formation.

In [**66**] a novel method of differentiate between short and long blinks was introduced . The system is intended to provide an alternate input modality to allow people with severe disabilities to access a computer. Images at the current and previous time stamps were taken and the bidirectional image is formed to detect motion and brightness changes in the scene to obtain eye candidates. This is used to detect the location of the eyes. Correlation scores for the user's eye at each frame of 'Open' and 'Closed' eyes were computed to detect blinks. This method recorded an accuracy of 95.6% but a task of spelling 2 words containing 8 letters took about 95 seconds. These image processing techniques are not robust and are prone to failure under limited lighting conditions. The real-time accuracy of these approaches is also low. Moreover, these approaches require the patient to learn the Morse coding scheme to communicate and interpret the messages.

An efficient system for eye blink detection is presented in [**64**]. This method uses Haar-cascade classifiers for the detection of the face and the eye positions relative to the face. An eye blinking detection based on eyelids state ('Open' or 'Closed') is used for controlling android mobile phones. The application was used in different environments to study the effects of varying lighting conditions and distances from the eye. The performance of Haar-cascade classifiers is not completely invariant to the change in lighting conditions and hence there is a decay in performance. Artificial light was used and an overall accuracy of 98% was obtained.

A low-cost implementation [**67**] of an eye-blink-based communication aid for ALS patients is presented in this work. The position of the pupil is found by generating four rectangles around it. Template matching is used to track the eye and detect eye blinks using hierarchical optical flow. The optical-flow method used here is also dependent on various assumptions (1) the brightness of any feature point is constant over time and (2) nearby points in the image move in a similar manner. The implementation, though being a low-cost implementation, has an accuracy of 94.75% during the typing test. Though the accuracy is comparable, the algorithm takes a lot of time to type a phrase. It takes approximately 2 seconds for the algorithm to generate a single scan of the eye which is a lot considering it is for a single character.

Multiple methods dealing with the detection of the eye state have been implemented in [**68**]. One of the methods is based on ideogram selection by moving a cursor on the screen. But this requires the user's gaze to be focused and not move around since that might lead to false positives. The second method is based on image binarization using an adaptive threshold determined using the integral sum image for the selection of ideograms using voluntary eye blink detection which is defined as a longer blinking interval than a normal physiological one.

Though this method seems user-friendly, as opposed to the first method, it requires a high learning curve as the user will have to learn the system in torder to determine the duration of blinks necessary for the ideogram selection. Considering the stated limitations and accuracy of 91.04%, this method does not seem feasible. A vision-based human-computer interface [**69**] is presented in the work. The interface detects eye-blinks and interprets them as control commands. A robust system that can detect faces and eyes using Haar-Cascade filters and can detect and interpret blinks using template matching with a fixed threshold is presented. An accuracy of 95.35% is achieved but the algorithm takes approximately 12 seconds to complete the process of detecting blinks which is considerably longer if communication needs to be carried out in real-time. This work [**62**] presents a real-time detection and classification between eye blink, left and right wink. The process of blink detection has been divided into four parts. The first part is face localization in facial images acquired through a video camera. Following that, eye pair localization, pixels' motion analysis using optical flow technique, and classification of eye blinks were performed. This approach was accurate 96, 92 and 88% of the time for detection of eye blink, left wink and a right wink, respectively. The latency for the detection of a single blink was found to be 250ms.

A wearable device [**63**] to detect eye blinks for alleviating dry eyes and computer vision syndrome was proposed in this work. The prototypes sense infrared reflections from the cornea of the person of interest while blinking. Eye blinks are evaluated using discontinuity in infrared reflections. This method [**63**] captured 85.2% of all the blinks that occurred during testing. The IR sensors tend to show false readings when the orientations are altered and hence unreliable to use in realistic scenarios. Any facial movements such as laughing, talking and yawning can also induce errors.

4.3. Methods

The goal of this approach is to detect if the person of interest blinks their eyes and to map the sequence of blinks to a particular entry in the dictionary of phrases. In order to achieve this, one has to detect the state of eyes (‘Open‘ or ‘Closed‘). If an ‘Open’ state is followed by a ‘Closed’ state in the subsequent frames, the system detects an eye blink. The system is divided into three phases as shown in Fig. 4.1.

Phase one: preprocessing data by capturing and saving the stream of images. Phase two: using the CNN model that we trained to predict the state of user’s eyes which includes training a convolutional neural network on the labeled dataset to detect the state of user’s eyes. Phase three: using the output vector of the predictor, map the eyeblinks to a phrase in the dictionary. In the

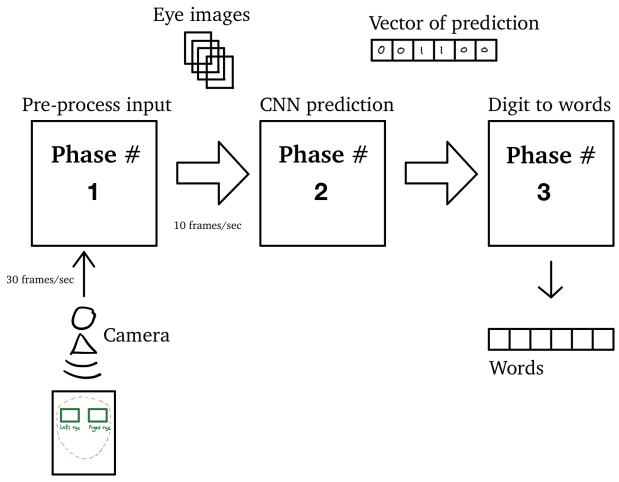


FIGURE 4.1. The 3 phases of the BWCNN system.

phase one, the system pre-processes the data by capturing and saving the stream of frames. The frame is captured by the camera (PC, laptop, or IoT) and the stream of frames is saved as image files. In order to improve performance, the system only saves a designated eye area. This step is important to reduce the dimensionality of the saved images which will be fed as an input to the Neural Network in the next phase. By saving only the eyes area, the size is reduced from 10KB to 3KB. The size of the image can be further reduced from 3KB to 2KB by converting the images to grayscale.

TABLE 4.1. Dictionary

# of blinks	Vector	e.g. Words	e.g. Moves
1	000000010000000	Yes	Right
2	000000101000000	No	Left
3	000001010100000	Hi	Click
4	000010101010000	I am	Up
5	000101010101000	Good	Down
6	001010101010100	Thanks	Exit
7	010101010101010	How are you?	Help

In phase two, the system predicts the contents of the image. Based on the image obtained in the previous phase, the state of the eyes is predicted. In this phase, we deal with the prediction of the image using a trained ConvNet. The fundamental challenge with deep learning is striking the right balance between generalization and optimization. The main question here is: given our dataset, what is the best CNN architecture to implement? Since this system runs in real-time, the latency (time taken for prediction) of the system is a very important factor. Thus, improving the accuracy of the prediction is not the only priority. The tradeoff between the accuracy and the latency (prediction time) will have to be taken into account. Moving forward, the question is, given our dataset, what is the best CNN architecture to implement in terms of accuracy and latency? In the coming subsection, we explain in detail the procedure we follow to choose the best architecture that we believe serves our goal. The input of this phase is one eye image and the output is a binary digit of zero or one. Zero represents the 'Open' state and the one represents the 'Closed' state.

In the phase three, the system processes the output and maps it to the phrases. The system stores the output of the CNN as a vector of zeros and ones. Each blink is represented by 010. Based on the output vector, the number of blinks are calculated and is mapped with the phrases in the dictionary.

4.3.1. Phase 1: Preprocessing data by capturing and saving a stream of images. In this section, we present the method of obtaining the data and preprocessing it to be given as input to the ConvNet.

4.3.1.1. *System Input.* The system uses camera devices (PC, laptop, IoT) for capturing the frames. Regulars webcams are capable of capturing 30 frames per second. Choosing the right number of frames per second will directly affect the user experience. Since, The system runs in

real-time, it is more effective to reduce the latency and not let the user wait for the results. At the same time using a small number of frames per second will lead to missing the eye blink and as a result, the CNN model mispredicts the state. Our experiment shows that 10 frames per second is a reasonable frame rate. So there is a frame being captured every 100ms, and in order for our system to avoid delay and operate in real-time, we need our model to predict each frame in less than 100ms. We further impose a constraint on the user that the 'Closed' state should be maintained for at least 200ms, so that two frames of that state are obtained for prediction. The first is the main prediction and the second is the redundant prediction which will be useful in case the user increases the blink rate or any other issue that leads to missing the current frame capture. Choosing the right architecture that has high accuracy and low latency is very critical. Thus, our trade-off here is choosing the right number of frames that serve the system without causing a huge delay. The system saves each frame as an Image of dimension 80X70 pixels and a size of 2KB and gray color. By converting them to grayscale, we reduced the frame size file from 3KB to 2KB.

4.3.2. Phase 2: Predict the contents of the image. In this section, we present the experiments to choose the best fitting neural network architecture for our application. Choosing the neural network architecture for a given data is a challenging task. The goal of our experiment here is to select the best-suited architecture among the four state-of-the-art architectures (SqueezNet, ResNet, InceptionV3 and DenseNet architecture). Not only the architecture but also the hyperparameters play a huge role in the model performance. The hyperparameter that we have chosen here is the batch size. We start by training the networks from scratch for different values of batch sizes and find the batch size that gives the best results. On finding the best batch size, we further explore the chances of improving the performance by using transfer learning with a pre-trained model and compare the results. By considering the trade-off between the accuracy and the time taken for prediction, we decide the best architecture that fits our system.

4.3.2.1. *Training Dataset.* We used the eye dataset from Media Research Lab (MRL) which is available for public use. The dataset contains 84,898 pictures of eyes taken from thirty-seven individuals consisting of thirty-three men and four women. Each image in the dataset was collected from one of the following sensors: Intel RealSense RS 300 sensor with a resolution of 640 x 480, IDS Imaging sensor with a resolution of 1280 x 1024, and Aptina sensor with a resolution of 752

x 480. The original dataset contains 6 different classes: 'gender', 'glasses', 'eye state', 'reflections', 'lighting conditions' and, 'sensor resolution'. But we do not need all these classes for our purpose. Therefore, we divided the entire dataset into just two classes - 'Open' and 'Closed'. We split the dataset into training and test sets. The training set consists of 80% of the images amounting to 67,919 images and the test set has 16,979 images which are 20% of the total number. Both the training and test set has an 'Open' and a 'Closed' class. The training directory thus has two folders named 'Open' and 'Closed' and likewise for the test directory too. The folder names of the images act as the labels for each image in that folder. You can download our dataset from here: http://albara.ramli.net/bwcnn/imx.zip

4.3.2.2. *Training experments.* Training the model is crucial to obtain an accurate detection of the eye's state. Apart from the accuracy, another important factor to consider is the latency for detection i.e., the time taken to make an accurate classification. The robustness of a model is measured based on these two factors and it is important to choose the right dataset and architecture for this purpose.

Deep learning is a field that is researched extensively. With that being said, there are a lot of potential algorithms and architectures that can help with our goal. Each of them has its own set of advantages and limitations and all of them tend to act differently with unseen data. Apart from the architecture, the hyperparameters used for training such as batch sizes play a huge role in the accuracy of the model. To find the model which can provide the best accuracy with the least latency, we implemented various state of the art architectures such as SqueezeNet, ResNet, InceptionV3 and DenseNet. We trained them from scratch by randomly initializing the weights. We can make a fair comparison of these architectures if they are trained with the same batch size and for the same number of epochs. We also used the weights from the trained model to train the same architecture with different batch sizes. Based on how each model performs on the test set, we can decide on which architecture suits our goal the best. The following are the experiments that were performed to analyze the neural networks:

Train ResNet architecture from scratch. There are a number of hyperparameters that can be tuned during the training. One of the hyperparameters is the batch size. We train the ResNet architecture for 100 epochs using 6 different batch sizes on our dataset and calculate the overall

accuracy. The performance metric used to analyze each batch size is the overall accuracy. Comparing the performance, we select the three best batch sizes. For our experiment, we tried batch size numbers 1, 2, 4, 8, 16 and 32. The results (Table. 4.2) show that using batch sizes 8, 16 and 32 provides the best accuracy among all the different batch sizes.

Run 500 epochs. Based on our observation of this experiment. After finding the best batch sizes, we wanted to further improve the performance of our network. Since our experiment stops only at 100 epochs, training the network for more number of epochs might improve the performance. Training the network for all the batch sizes is computationally expensive, so the residual network is trained only on the three selected batches for 500 epochs. The obtained results (Table. 4.2) show that there is no further improvement in terms of accuracy.

Transfer learning. Another way of improving network performance would be to use transfer learning. Here our pre-trained model of the ResNet is used as the source network. The best weights obtained from the three selected batch sizes 8, 16, and 32, are used to train the source network. After 100 epochs of training, the results (Table. 4.2) show no improvement in the accuracy beyond what we have already obtained in the previous step. We also implemented transfer learning using a pre-trained model of ResNet50 as our source network and training it on the same set of three different batch sizes as before. Similar to the previous results, after 100 epochs the there was no significant improvement in the accuracy of the network. (Table. 4.2)

Train Inception, SqueezNet and DeneseNet architectures from scratch. After implementing the different variations of ResNet, the next approach towards finding the best network would be to try different architectures. Trying different batch sizes and testing every case is very expensive. Thus, our assumption is that the same set of 3 batch sizes from ResNet would be the best performing batch sizes in the other architectures as well. To investigate this assumption we run the same experiment again but with different architectures. DenseNet, Inception, and SqueezeNet are the three different architectures that were chosen for this purpose. (Table. 4.3). The time taken for prediction for each frame is calculated for the best performing batch size for each architecture. This is a measure of the latency of the networks. The comparison is made between each architecture based on the accuracy and the latency.

TABLE 4.2. ResNet with and without transfer learning

		TRANSFER LEARNING FROM BATCH SIZE 16		TRANSFER LEARNING FROM OFFICIAL RESNET50		BEST 3 BATCH SIZES FOR 500 EPOCHS		
Batch Size	No. pf epochs	Accuracy	Last ep. improved	Accuracy	Last ep. improved	No. pf epochs	Accuracy	Last ep. improved
8	100	99.22	31	99.22	29	500	99.20	60
16	100	99.23	51	99.17	16	500	99.21	33
32	100	99.22	49	99.17	25	500	99.19	46

TABLE 4.3. DenseNet,SqueezeNet,InceptionV3

		DENSENET		SQUEEZENET		INCEPTIONV3	
Batch Size	No. pf epochs	Accuracy (%)	Last ep. improved	Accuracy (%)	Last ep. improved	Accuracy (%)	Last ep. improved
8	100	99.24	55	49.40	1	99.14	35
16	100	99.18	70	49.40	1	99.20	22
32	100	99.21	52	49.40	1	99.17	38

4.3.2.3. *Testing the Model.* The first phase of our system is to obtain the dataset of images containing the two eye states, 'Open' and 'Closed' and these images are preprocessed to make them apt for the contiguous steps. Following the preprocessing phase, we move on to the next phase, detection. In this step, given an input image, we predict the state of the eyes by using one of the above-stated pre-trained networks. The network classifies them as zeros (for 'Open' state) and ones (for 'Closed' state). The third and final phase involves detecting eye blinks based on the eye state predictions and mapping it to a sequence of words.

4.3.3. Phase 3: Mapping. The output of the neural network is a binary classification. The system stores the output of the network as a vector of zeros and ones. In this phase, the system normalizes all the changing state to one edge. For example, the vector 00000110000 becomes 010, where one represents the image of a 'Closed' eye and zero represents the image of an 'Open' eye. The change in state is considered as an edge and all the redundant states do not matter. Each blink is represented by 010. Based on the output vector which contains a representation of the blink sequence, the number of blinks is calculated and is mapped with the phrases in the dictionary. We have a previously defined dictionary of phrases. These are basic phrases that we use in everyday life and the dictionary can be further increased with more phrases. The dictionary maps every phrase with a certain vector of zeros and ones. (See Table. 4.1)

4.4. Results

This section provides the performance results obtained from different architectures. Table. 4.1 discusses the results obtained from training ResNet for 100 epochs on different batch sizes (1,2,4,8,16,32).

TABLE 4.4. Final Results

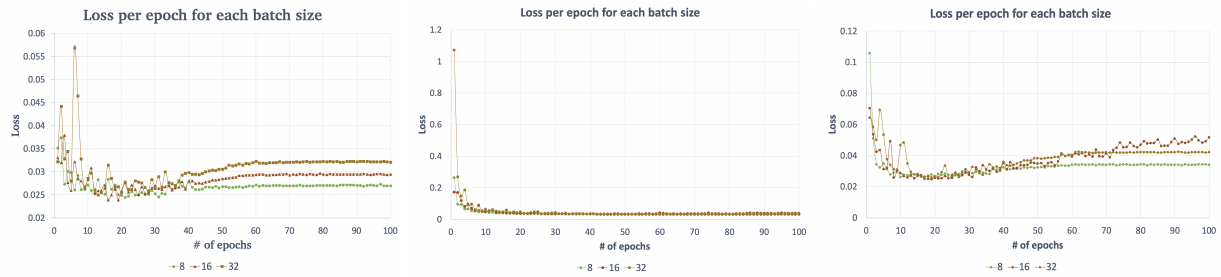
Architecture	*ResNet*	*DenseNet*	*SqueezeNet*	*InceptionV3*
Batch size	*16*	*8*	*16*	*16*
Last imp. epoch	*55*	*55*	*1*	*22*
Total params	*23,591,810*	*7,039,554*	*723,522*	*21,806,882*
Trainable params	*6,955,906*	*6,955,906*	*723,522*	*21,772,450*
Non-trainable params	*83,648*	*83,648*	*0*	*34,432*
Model size	*283MB*	*85MB*	*8MB*	*262MB*
Accuracy	99.26%	99.24%	49.40%	99.20%
Avg latency	117.28ms	146.09ms	13.64ms	94.1ms

The results show the accuracy obtained for each batch and the last epoch where there was an improvement in the accuracy. It can be seen that training the network for 100 epochs on a batch size of 16 yields the best accuracy of 99.26%. It can also be seen that there is no improvement in accuracy after 55 epochs.

Table. 4.2 shows these results for 500 epochs for the three best batch sizes (8,16,32). Our trained ResNet model was trained again using transfer learning, by initializing the weights obtained from our model. We also used transfer learning to train ResNet50 for our dataset by initializing it with our weights. These results are shown in Table. 4.2 and Table. 4.2 respectively. Table. 4.2, Table. 4.3 presents the results for different architectures. Table. 4.2 encompasses the same category of results for the DenseNet architecture for the three batch sizes (8,16,32) that were experimentally determined to be best for ResNet.

Fig. 4.2 shows the loss curve for this. DenseNet has the best accuracy of 99.24% when trained for 100 epochs on a batch size of 8. The results obtained from training the SqueezeNet and InceptionV3 architectures are presented in the table. SqueezeNet seems to be the least performing among all the architectures with an accuracy of 49.40% for 100 epochs and a batch size of 16, and also there is not a change in the accuracy after the first epoch which is evident from Fig. 4.2. The final comparison between the best results obtained from the different architectures can be seen in Table. 4.4. We can see that InceptionV3, DenseNet and ResNet has similar accuracies but the InceptionV3 model has the lowest prediction time so we implement the InceptionV3 to predict the state of the system.

FIGURE 4.2



(a) ResNet validation loss curves for ResNet transfer learning to itself from pre-trained of batch size 16 to batch sizes 8, 16, and 32.

(b) DenseNet validation loss curves of batch sizes 8, 16, and 32.

(c) InceptionV3 validation loss curves of batch sizes 8, 16, and 32.

Apart from the accuracy of the networks this table also contains the time (in milliseconds) required for the prediction of each frame of image for different architectures for our application. We can see that ResNet has the best accuracy of the lot (99.26%), but has a high latency of 117.28ms. InceptionV3 has an accuracy of 99.20% which is close to ResNet but with a lower latency of 94.1ms. Fig. 4.2 shows the validation loss curve for InceptionV3 of batch sizes 8, 16, and 32. The trade-off between accuracy and latency need to be considered to select a network for any application. Thus, the overall accuracy of the system is 99.20% and the latency is 94ms.

4.5. Discussion

We trained different architectures with different hyperparameters to identify which combination gives the best accuracy with the lowest latency. This is our basis for comparison to find the right model for our application. To make it clear, our priority is to obtain the highest accuracy on the test set and if the accuracies are comparable, we can compare the latencies and choose the one with the lowest latency. For the sake of conducting a clear comparative analysis, we are comparing the results of each architecture for a constant batch size of 8. SqueezeNet received the lowest accuracy with the least number of parameters. The DenseNet, ResNet and, Inception-v3 acquired accuracies in the same range of 99.20% and above. Since they have similar accuracies, it makes sense to compare their latencies since our application must work in real time. On making this comparison,

we found that the Inception -v3 acquired the least latency, making it the appropriate model for this approach.

ResNet which is short for Residual network, gave us the best accuracy and had the greatest number of parameters. It also had the largest number of layers. Since it has a very deep network, it also must deal with the vanishing gradient problem. ResNet tackles this problem by creating several "identity shortcut connections" that skips one or more layers which means that there is a path from one layer to another. Since there is a direct connection between one layer to another [**70**], during forward propagation, integrity of information can be protected by directly sending the input to the next layer through shortcut paths. Also, during backpropagation, since there are shortcut paths, the gradients have to move through lesser layers than usual which reduces the vanishing gradient problem significantly. So, based on our understanding of ResNet, we believe that the shortcut paths allowed to improve the accuracy of prediction. We introduced transfer learning for the ResNet architecture, in the hopes of improving the accuracy and latency. We used the weights trained on the official ResNet50. We see that transfer learning does not help us in improving the testing accuracy and latency, but it did help us in converging to optimality much more easily than by randomly initializing the weights.

DenseNet breaks away from the stereotype of deepening network layers and widening network architecture to improve network performance. Through feature reuse and Bypass setting, it can not only greatly reduce the number of parameters in the network, but also alleviate the emergence of the gradient vanishing problem [**71**]. DenseNet has several advantages. Firstly, it could reduce gradient vanishing problems. Secondly, it could enhance the spread of features. Thirdly, feature reuse is encouraged in this network architecture. Finally, it also could reduce the number of entries. In this network, there is a direct connection between any two layers, which means that the input of each layer of the network is the union of the output of all the previous layers, and the feature graph learned by this layer will be directly passed to all the subsequent layers as input. This explains how the model reaches very high accuracy with this architecture. They have designed the output channel in such a way that the convolution layer is very small. We think that this acts as a bottleneck and thus increases the time for prediction.

Since latency is an important concern for our approach too, we worked on the Squeezenet architecture in the aim of reducing the prediction time required to less than 100ms. The main idea behind SqueezeNet is to use 1x1 pointwise filters replacing 3x3 filters. The building brick of SqueezeNet is called the 'fire module' which contains a Squeeze layer and an expand layer. SqueezeNet stacks a bunch of fire modules along with a few pooling layers. The Squeeze layer reduces the depth to a small number while the expand layer increases it [**72**]. They both act together to keep the feature map to be of the same size at the end. We believe that the squeezeNet did not receive a high accuracy because it does not have a deep network and has a very low number of parameters to train on. All mainstream Deep learning networks have large number of layers. Large number of layers tend to increase the accuracy, but this improved accuracy is accompanied by several disadvantages. Firstly, very deep networks are prone to overfitting and if the training set is limited, it is difficult to generalize to any application. Also, the larger the network is the greater the computational complexity. So, it is difficult to implement that type of network practically. Thirdly, we also have the vanishing gradient problem which we must deal with for very large networks. It gets extremely difficult to optimize the cost function.

Inception network works in increasing the depth and width of the network while also reducing the number of parameters to be trained. This is the reason why Inception has a prediction time lesser than all the other architectures. Inception version 3 transfers all the features of the previous versions along with its own distinctive characteristics. In the inception -v1, the sparse CNN is approximated with a dense construction. Since only a small number of neurons are effective, the kernel size is made smaller to make the same computations but in a less expensive manner. It uses convolutions of varied sizes and scales to capture the different features. It uses the scales which can reduce the computational cost [**73**]. This is another reason why Inception gives out a lower latency. Another salient point is that the Inception network has a bottleneck layer (1x1 convolutions) which can reduce the dimensionality immensely just saving up on computation power and time.

InceptionV2 is an improvement to the first version where they introduced factorization. By factorization what they mean is that they have reduced the kernel size even more to make them computationally cheap. They have altered the 7x7 convolutions into two 3x3 convolutions since 7x7 convolutions are 2.78 times computationally expensive than 3x3. After that they have changed the

3x3 convolutions into a pair of 1x3 convolutions and 3x1 convolutions. This makes the computation 33% faster [**74**]. This feature is present in the final version of inception too and we believe this is another reason for the low latency that we achieved in our system. Adding to this feature, Inception -v3 also has batch normalization in the auxiliary classifiers, RMSprop optimizer and Label smoothing which would have contributed to the fast training and good accuracy.

4.6. Conclusion

In this work, we designed a system for ALS patients which could convert eyes blinks to word with Convolutional Neural Network (CNN). The system uses a CNN to predict the state of the eyes of the person of interest and this was used to find the blinking pattern. We compared several CNN architectures and discussed hyperparameter selection in model training. For the evaluation, we tested our system using 16,979 facial images in our testing dataset and found that our proposed prediction model was efficient and effective. Results demonstrate that overall prediction accuracy is 99.20% and an average prediction time is 94ms with InceptionV3. In our future work, we plan to improve the eye detection and prediction speed. Maybe we could use a single network for detection and prediction to reduce the response time. We also plan to add more complex language models to better serve those who need this system.

CHAPTER 5

MASTAF: A Model-Agnostic Spatio-Temporal Attention Fusion Network for Few-shot Video Classification

We propose MASTAF, a Model-Agnostic Spatio-Temporal Attention Fusion network for few-shot video classification, to solve the limited number of video samples in real-life HAR applications, a common challenge shared with computer vision. MASTAF takes input from a general video spatial and temporal representation,e.g., using 2D CNN, 3D CNN, and video Transformer. Then, to make the most of such representations, we use self- and cross-attention models to highlight the critical spatio-temporal region to increase the inter-class distance and decrease the intra-class distance. Last, MASTAF applies a lightweight fusion network and a nearest neighbor classifier to classify each query video. We demonstrate that MASTAF improves the state-of-the-art performance on three few-shot video classification benchmarks(UCF101, HMDB51, and Something-Something-V2), e.g., by up to 91.6%, 69.5%, and 60.7% for five-way one-shot video classification, respectively.

5.1. Introduction

Few-shot learning has received increasing attention in video classification for its potential to reduce the video annotation cost significantly [**75**]. In few-shot video classification, the video samples in the training and test sets are from different classes (i.e., unseen classes in the test set). To classify an unlabeled video sample (query), a few-shot video classification model aims to classify the query to the unseen class (support set). Inspired by the development in few-shot image classification [**76**,**77**,**78**], recent few-shot video classification approaches using metric-learning-based methods achieve state-of-the-art performance [**75**,**79**]. This work targets metric-learning-based few-shot video classification.

A metric-learning-based few-shot video learning algorithm classifies a query based on the similarity between the representation of the query video and the representation of each class in the

support set. Therefore, the core to metric-learning-based few-shot video classification is to design feature extraction and representation for the support sets and the query. Many feature embedding networks have been designed for this purpose. Perrett [**79**] leverages attention mechanism in temporally-ordered frames from support sets to match query frames after extracting representation for each frame with pre-trained 2D Convolutional Neural Network(2D CNN). Zhang [**80**] introduces permutation-invariant pooling and self-supervised learning tasks to enhance representations after extracting from a 3D Convolutional Neural Network(3D CNN) embedding network.

In few-shot scenarios, prior efforts with a 2D CNN embedding network outperformed those with a 3D CNN embedding network [**75**, **79**, **81**]. However, there are two considerable limitations in existing work with a 2D CNN embedding network. The first limitation is that a complex temporal alignment strategy between the video frames for better accuracy increases computational demand and model inference runtime. For example, Perrett [**79**] achieves SOTA performance on few-shot video classification by exploring all the combinations of two and three ordered sampled frames from a video for temporal information. As the number of sampled frames from a video grows, the computational cost and inference runtime increase significantly.

The second limitation is their inability to maintain high performance when replacing a 2D CNN embedding network with other advanced video representation models such as 3D CNN [**82**, **83**, **84**, **85**] and Video Transformer [**86**]. With the release of large-scale video datasets, video classification models' performance based on 3D CNN and Video Transformer surpasses those with 2D CNN [**82**, **86**, **87**], which means such models can generate a better representation for discrimination. Therefore, one would expect that if we replace the 2D CNN in the existing few-shot video classification models with an advanced video representation model, performance should improve. However, this did not happen. Instead, Zhu [**81**] found that 3D CNN models [**82**, **84**, **85**] do not perform better than 2D CNN models in PAL [**81**], a SOTA 2D-CNN based few-shot video classification algorithm. The main reason is that 2D-CNN approaches rely on the frame-level similarity score and temporal alignment, which do not exist in a 3D CNN embedding network.

In this work, we propose a model-agnostic few-shot video learning algorithm named Model-Agnostic Spatio-Temporal Attention Fusion network(MASTAF). Our key motivation is to make

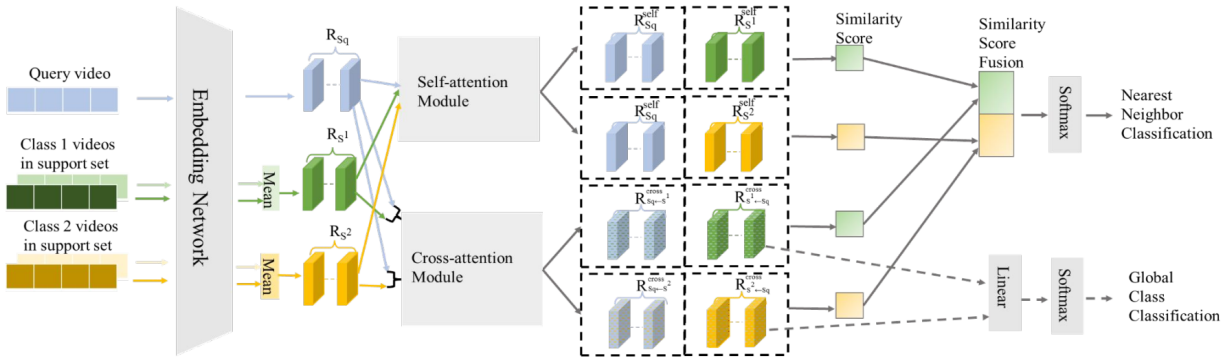


FIGURE 5.1. Illustration of the Model-Agnostic Spatio-Temporal Attention Fusion(MASTAF) on a 2-way 2-shot video classification. First, we extract spatio-temporal features with a pre-trained embedding network for each video. Then, we compute a prototypical representation(R_{S^c}) for each class in the support set, which is the mean of all the representations of each class. After that, we use the self-attention module to highlight spatio-temporal features for each query and support class representation and compute the similarity score of each pair of query representation and support class representation using cosine distance. In parallel, we use the cross-attention module to highlight the spatio-temporal correlation features for each pair of query representation and support class representation, and compute the similarity score using the cosine distance. The cross-attention representations of each class in the support set are fed into a global video classifier as a multi-task training set. And the fusion results of similarity scores from the self-attention module and cross-attention module are fed into the nearest neighbor classifier. Details are in Section 6.3.

the most of the rapid advances in video representation learning to build a simple and effective few-shot video learning framework. To achieve this goal, we have to address the limitations discussed above. Advanced video representation networks, such as 3D CNN and Transformer, extract spatio-temporal representations directly instead of frame-level information. To make good use of such representations, we use self- and cross-attention models to increase the weight of the critical spatio-temporal region to increase the inter-class distance and decrease the intra-class distance as shown in Figure 6.1. The self-attention network emphasizes the regions of the representation that are essential for representing each class and the query, while the cross-attention network emphasizes the regions of the representation that enhance the discriminability between the query and the unseen classes in the support set. Then, we measure the similarity between the query and each unseen class based on the feature maps from each attention network. Last, we classify the query video by a simple yet effective fusion network. We also add one multi-task training setting,i.e., global

video classification task, to regularize the embedding module and further improve generalization performance. More details are presented in Section 6.3

Contributions We make the following contributions.

1.We propose MASTAF, a simple and effective attention-based network compatible with different video classification models for few-shot video classification. MASTAF can benefit from advanced video classification models such as 3D CNN and video Transformer that extract good spatial-temporal representations.

2.We design a fusion mechanism to integrate self-attention and cross-attention networks, which greatly enhances the essential spatial and temporal regions of video representation.

3.We extensively evaluate MASTAF using three benchmarks, i.e., UCF101 [**88**], HMDB51 [**89**], and Something-Something V2 [**90**]. Compared to the existing work, MASTAF achieves state-of-the-art performance with a 2D CNN embedding network and improves the state-of-the-art performance with a 3D CNN embedding network without additional computational cost. Our code is available at `https://anonymous.4open.science/r/STAF-30CF`.

5.2. Related work

Few shot learning Most existing few-shot learning algorithms can be divided into three categories: model-based methods [**91**, **92**], optimization-based methods [**?**, **?**], and metric-learning-based methods [**75**, **79**, **93**]. Metric-learning-based methods are more promising than other two methods in few-shot video classification since the previous work with metric-learning-based achieved better performance [**75**, **79**].

Metric-learning-based method measures the distance between the representation of support samples and query samples and classifies them with the aid of the nearest neighbor to keep similar classes close and dissimilar classes far away. Particularly, Prototypcal Network [**93**] is based on the idea that each class has a Prototypical representation which is the mean value of support set in embedding space. The few-shot learning problem then becomes the nearest neighbor in the embedding space. Our work is one of the metric-learning-based methods. We can take input from general video spatial and temporal representations extracted from different video representation

models. To make the most of representations in the embedding space, we highlight the spatio-temporal features that need attention for each class while increasing the differences from other classes.

Few shot video classification The first module of most metric-learning based methods for a few-shot video classification model is an embedding network that extracts features from each video. The two most commonly used approaches for embedding network are 2D CNN [**75**, **79**, **81**, **94**, **95**] and 3D CNN [**80**, **96**, **97**]. After using 2D CNN to extract features from each video frame, Zhu and Yang [**94**, **95**] introduce a memory network structure to learn optimal representations in a larger video representation space. Instead of creating a memory structure to memorize long-term information for video representation, more recent work with 2D CNN embedding networks focus on temporal alignment exploration between the query video and the support set. Cao [**75**] aligns the frames between the query video and support video by temporal ordering information. Perrett [**79**] achieves SOTA on 5-way 5-shot video learning by computing the distance of temporal-relational representations between each frame of query video and support video. In comparison, the features extracted from the 3D CNN embedding network already contain temporal information. Therefore, recent work focus on generating general spatio-temporal video representations for unseen classes. Dwivedi [**97**] leverages GAN to generate the spatio-temporal video representations for the prototype of the unseen classes. Zhang [**80**] introduces permutation-invariant pooling and self-supervised learning tasks to enhance representations whereas Bishay [**96**] uses segment-based attention and deep metric learning. Recently, video Transformers have become a promising option for video representation due to their long-term reasoning ability [**86**, **98**]. Although video Transformers are not widely used in few-shot video classification, SOTA performance in video classification indicates the promise of being applied in a few-shot scenario.

Attention-based learning Attention mechanism enhances the learning ability of long-range dependencies in the network to highlight the critical regions of visual representations [**99**]. These critical regions are useful in discriminating the differences between different classes. Therefore, the recent work with attention mechanisms achieve SOTA accuracy for few-shot learning tasks [**77**, **79**].

Hou [**77**] leverages the cross attention mechanism to extract discriminative representations for few-shot image classification. While for few-shot video classification, Perrett [**79**] applies a cross transformer with a multi-head attention mechanism for the representation of each frame to locate the representative frames for similarity computation.These works adopt 2D CNN to extract the features and apply frame-level attention mechanisms for temporal alignment. However, these works cannot maintain high performance when using a 3D CNN embedding network without the help from complex temporal alignment. Our work is compatible with any video classification model and uses an attention fusion network to highlight the spatio-temporal features, which help increase the inter-class distance and decrease the intra-class distance.

5.3. MASTAF: Model-Agnostic Spatio-Temporal Attention Fusion Network

5.3.1. Problem definition. The few-shot video classification problem aims to classify one unannotated query video into one of several annotated categories set, which we call "support set". Each category has only a few video instances in this support set, and the model did not see these categories during the training process. Our work focuses on C-way K-shot video classification, where C denotes the number of categories in the support set and K represents the number of video instances for each category in the support set. We follow the same episodic training as in the previous study [**75**, **79**, **80**, **94**, **95**] that randomly select C classes with K video clips for the support set. Then we select one query video from these C classes, which is different from the K video clips in the support set. For each C-way K-shot episode, the support set contains C classes, and each class has K video clips.

We use $S_k^c = \{f_{k,1}^c, f_{k,2}^c, \ldots, f_{k,n}^c\}$ to denote the k^{th} video clip of class c, where c belongs to C and k belongs to K, $f_{k,i}^c$ denotes the i^{th} extracted frame from the video and n denotes the total number of frames extracted from the video. For the query video, we use $S_q = \{f_1, ..., f_i, ..., f_n\}$, where f_i denotes the i^{th} frame extracted from the query video and n denotes the total number of frames extracted from the query video. The final goal is to predict S_q to one of the classes.

5.3.2. The MASTAF Model. The design principle of the MASTAF model is to highlight the critical spatio-temporal region to minimize the intra-class distance while maximizing the inter-class distance between the query video and support set. To tackle the challenge of only having few

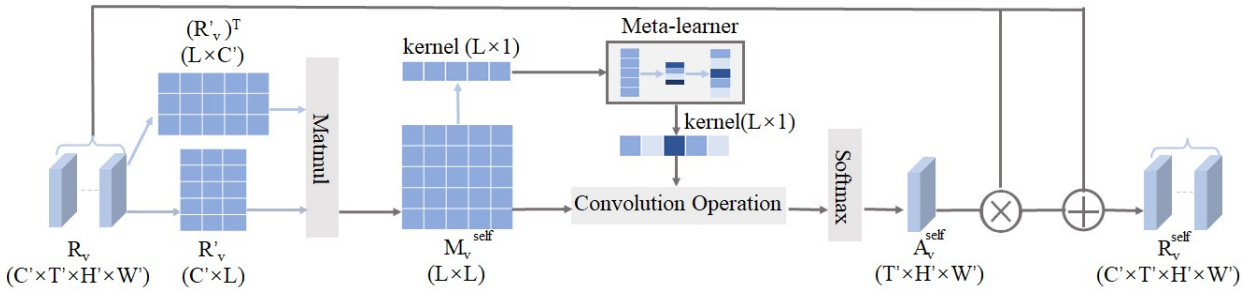


FIGURE 5.2. Self-attention module

samples for the unseen class, we first extract spatio-temporal features using any video classification model. Then, we use the attention fusion module to further highlight the critical spatio-temporal region for metric learning. In parallel, we use a global classification task to regularize the embedding network. Next, we analyze each module in the MASTAF model, which is described in Figure 6.1.

Embedding module In the MASTAF model, the goal of the embedding module f_φ is to learn the spatio-temporal representations for each video. We evenly extract frames from each video, where n is the total number of frames extracted from each video. We can use any video classification model as the spatio-temporal embedding module.

Given a frame sequence extracted from the video $S_v = \{f_1, f_2, \ldots, f_n\}$, let $R_v \in \mathbb{R}^{C' \times T' \times H' \times W'}$ denote the representation learned from the embedding model:

$$R_v = f_\varphi(S_v). \tag{5.1}$$

For a video clip in the support set S_k^c, we use $R_{S_k^c}$ to denote the representation learned from the embedding module. We use R_{S^c} to denote the representation of the class c, which is the mean of all the representations of video clips for class c in the support set. And since we have only one query video in the few-shot learning task, we use R_{S_q} to denote the representation for the query video clip. After we get the representations for the support set and query video, we go through two separate attention modules in parallel,.i.e, the self-attention module and the cross-attention module.

Self-attention module Our goal of the self-attention module is to highlight the critical information in the representation of each class. As shown in Figure 5.2, we first reshape each representation to $R_v^{'} \in \mathbb{R}^{C' \times L}$, where $L(L = T' \times H' \times W')$ is the number of spatio-temporal positions on each feature cubic map. After that, for each class in the support, R_{S^c} becomes $R_{S^c}^{'}$, i.e., $[R_1^{S^c}, \dots R_i^{S^c} \dots, R_L^{S^c}]$, where $R_i^{S^c}$ denotes the feature vectors at the i^{th} spatio-temporal position in the $R_{S^c}^{'}$. For each query video, R_{S_q} becomes $R_{S_q}^{'}$, i.e., $[R_1^{S_q}, \dots R_i^{S_q} \dots, R_L^{S_q}]$, where $R_i^{S_q}$ denotes the feature vectors at the i^{th} spatio-temporal position in the $R_{S_q}^{'}$. Then we compute the self-relation map for each representation as:

$$M^{self} = (R_v')R_v', \tag{5.2}$$

where $M^{self} \in \mathbb{R}^{L \times L}$ that denotes the self-relation map for each video, where M_i^{self} denotes the self-relation at the i^{th} spatio-temporal position in the feature map. Then we apply convolutional operation with a kernel d, i.e., $d \in \mathbb{R}^L$, to fuse each position self-relation vector into an attention scalar, which is in $\mathbb{R}^{T' \times H' \times W'}$. Then we leverage a softmax function to draw self-attention for each i^{th} position:

$$A_i^{self} = \frac{exp((dM_i^{self})/\tau)}{\sum_{j=1}^{L} exp((dM_j^{self})/\tau)}, \tag{5.3}$$

where τ is the temperature hyperparameter to amplify the variance and A_i^{self} denotes the i^{th} position of self-attention map A^{self},i.e., $A^{self} \in \mathbb{R}^{T' \times H' \times W'}$.

Instead of assigning equal weight to every position, we add a meta-learner to learn the kernel d dynamically to pay attention to the critical positions in the feature cubic map. First, we leverage row-wise global average pooling for M^{self} to get an averaged vector $\overline{M}^{self}$,which $\overline{M}^{self} \in \mathbb{R}^L$. Then we use a meta-learner to learn the kernel d dynamically:

$$d = f_\gamma(\sigma(f_\delta(\overline{M}^{self}))), \tag{5.4}$$

where $f_\delta : \mathbb{R}^L \to \mathbb{R}^l$ and $f_\gamma : \mathbb{R}^l \to \mathbb{R}^L$,.i.e, l denotes the scaled dimension and σ represents the ReLU function [**100**].

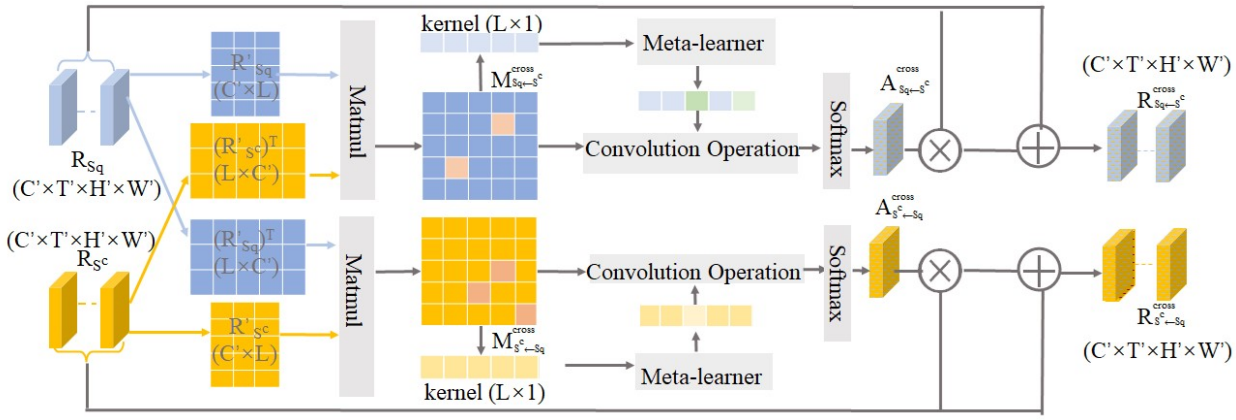


FIGURE 5.3. Cross-attention module

After we get the self-attention cubic map A^{self}, we leverage a residual attention mechanism to weigh each element of the original map R_v with $1 + A^{self}$ to get the self-attention representation R^{self} for each class:

$$R^{self} = R_v(1 + A^{self}), \tag{5.5}$$

where $R^{self} \in \mathbb{R}^{C' \times T' \times H' \times W'}$.

Cross-attention module While the self-attention module highlights the critical spatio-temporal region in the representation itself, the cross-attention module focuses on the correlation between the query video and the support set. As shown in Figure 6.2, we follow the same steps as in the self-attention module to reshape each representation to $R'_v \in \mathbb{R}^{C' \times L}$. After that, we compute the correlation map for each pair of the query video and the support class prototype. For example, for the pair of the query video R_{S_q} and support class c,i.e., R_{S^c}, we compute the correlation map for the query video $M^{cross}_{S_q \leftarrow S^c}$ between the query video and support class:

$$M^{cross}_{S_q \leftarrow S^c} = (R'_{S^c}) R'_{S_q}. \tag{5.6}$$

Then for the support class c, the correlation map $M^{cross}_{S^c \leftarrow S_q}$ between the query video and support class is:

$$M^{cross}_{S^c \leftarrow S_q} = (R'_{S_q}) R'_{S^c}. \tag{5.7}$$

After getting the correlation map for query video and support class in each pair, we go through the same steps as in the self-attention module, which are shown in the Figure 6.2, to get the cross-attention representation for query video and support class in each pair,i.e., $R^{cross}_{S_q \leftarrow S^c}$ and $R^{cross}_{S^c \leftarrow S_q}$.

Attention fusion module After we get the self-attention and cross-attention representation from the two attention modules, we compute the probability of predicting S_q as the class k using self-attention representation:

$$P_{self}(y = k|S_q) = \frac{exp(-D_{cos}(R^{self}_{S_q}, R^{self}_{S^k}))}{\sum_{j=1}^{C} exp(-D_{cos}(R^{self}_{S_q}, R^{self}_{S^j}))}, \tag{5.8}$$

where D_{cos} denotes the cosine distance and $P_{self}(y = k|S_q)$ denotes the probability of predicting S_q as the class $k \in \{1, 2, ..., C\}$ using self-attention representations. Then we compute the probability of predicting S_q as the class k using cross-attention representation:

$$P_{cross}(y = k|S_q) = \frac{exp(-D_{cos}(R^{cross}_{S_q \leftarrow S^k}, R^{cross}_{S^k \leftarrow S_q}))}{\sum_{j=1}^{C} exp(-D_{cos}(R^{cross}_{S_q \leftarrow S^j}, R^{cross}_{S^j \leftarrow S_q}))}, \tag{5.9}$$

where $P_{cross}(y = k|S_q)$ denotes the probability of predicting S_q as the class $k \in \{1, 2, ..., C\}$ using cross-attention module.

To take advantage of the discriminative information from two attention mechanisms, we leverage the attention fusion module with the nearest neighbor classifier:

$$P(y = k|S_q) = \frac{1}{2}[P_{self}(y = k|S_q) + P_{cross}(y = k|S_q)], \tag{5.10}$$

where $P(y = k|S_q)$ denotes the final probability of predicting S_q as the class $k \in \{1, 2, ..., C\}$.

Multi-task training To reduce the risk of overfitting in the training dataset and generate a general representation for unseen class, we train the MASTAF model in a multi-task setting to regularize the embedding network. We combine the nearest neighbor classifier and the global video classifier.

During the training process, after the attention fusion module computes the probability of predicting query video to one of the classes in the support set, we use a negative log-probability as the loss function of the nearest neighbor classifier based on the actual class label:

$$L_1 = -\sum_{k=1}^{C} logP(y = k|S_q). \tag{5.11}$$

Since the representations after the cross-attention module contain highlighting regions related to the query video, we choose these representations to predict the global class in the whole training dataset. The total class number in the training dataset is Z. We feed these cross-attention representations to a fully connected layer and a softmax layer to get the probability of predicting the global class,i.e., $P(y = z|S^c)$ where $z \in \{1, 2, ..., Z\}$. Then we define the loss function of the global video classifier as:

$$L_2 = -\sum_{z=1}^{Z} logP(y = z|S^c). \tag{5.12}$$

Finally, the loss function of the MASTAF model is defined as:

$$L = L_1 + \lambda\ L_2, \tag{5.13}$$

where we use λ to weigh the impact of different classification tasks. Note that a multi-task training setting is only used during the training process. This setting is discarded at the inference stage.

5.4. Evaluation

5.4.1. Experimental Setup. Datasets. We compare MASTAF with existing work on the UCF101 [**88**], HMDB51 [**89**], and Something-Something V2 (SSv2) [**90**]. We do not use Kinetics-100 [**94**] to avoid bias because one of our MASTAF models is pre-trained on Kinetics-700 [**101**]. In these datasets, SSv2 is more challenging because it focuses on actions related to temporal relationships such as 'pretending to take something from somewhere' versus 'take something from somewhere' [**102**]. There are two few-shot splits for SSv2 proposed by CMN [**94**] and OTAM [**75**], containing 64, 12, and 24 classes as the training, validation, and test set. We use SSv2-part and SSv2-all denote the split from CMN [**94**] and the split from OTAM [**75**]. The difference between these two splits is the number of video samples in each class. For SSv2-part, Zhu and Yang [**94**] randomly selects 100 samples for each class, whereas for SSv2-all, Cao [**75**] uses all the samples in the original SSv2. We evaluate our method in these two splits. Additionally, we also follow the split in ARN [**80**] for HMDB51 and UCF101.

Evaluation and baseline. Following the evaluation process in TRX [**79**], we evaluate the 5-way 1-shot and 5-way 5-shot video classification task and report the average accuracy over 10,000

randomly selected episodes from the test set. We compare our results with eight SOTA algorithms, i.e., TSN++ [**103**], CMN-J [**95**], OTAM [**75**], FEAT [**104**], PAL [**81**], TRX [**79**], ProtoGAN [**97**], ARN [**80**]. For a fair comparison, we use three MASTAF models with three different types of embedding networks, i.e., MASTAF-{TSN}, MASTAF-{R3D} and MASTAF-{ViViT}. For MASTAF-{TSN}, we follow the same embedding network configuration with [**75**, **79**, **81**, **104**], using an ImageNet pre-trained ResNet-50 as the backbone network. For MASTAF-{R3D}, we use the merged video dataset with Kinetics-700 [**101**],Moment-in-time [**105**], and START-action [**106**] to pre-train 3D ResNet-50 embedding network. We also compare our approach against the previous work based on a 2D CNN embedding network where we replace 2D CNN with 3D CNN. We use TRX-{R3D} as one of the baselines by replacing the 2D CNN embedding network with a 3D CNN embedding network(same pre-trained R3D model as MASTAF-{R3D}) in TRX [**79**]. We extract one representation using pre-trained R3D from each video and then go through temporal CrossTransformers proposed in TRX [**79**]. For MASTAF-{ViViT} and TRX-{ViViT}, we use ViViT [**86**] as our embedding network. We initialize ViViT from a ViT [**107**] image model trained on the JFT [**108**] dataset. Due to the huge computation demand for ViViT [**86**], we only perform 5-way 1-shot learning for MASTAF-{ViViT} and TRX-{ViViT}.

Experimental Configuration. For MASTAF-{TSN}, MASTAF-{ViViT} and TRX-{ViViT}, we evenly sample 8 frames from each video as 8 segments for each video. For 3D CNN-based MASTAF and TRX-{R3D}, we evenly sample 16 frames from each video sample. After that, we resize each frame to 256×256. Then we randomly flip each frame horizontally and crop the center region of 224×224 to augment the training data. For test data, we only crop the center with the same size without the horizontal flipping. Then for MASTAF-{TSN}, we use an ImageNet pre-trained ResNet-50 as the backbone and average all the frame representations as to the video representation. For 3D CNN-based MASTAF and TRX-{R3D}, we use a 3D ResNet-50 [**87**] with the weights pre-trained on the combined dataset with Kinetics-700 [**101**], Moments in Time [**105**], and Start Action [**106**] as the embedding network. After finetuning in the validation dataset, We set 0.025 as the temperature hyperparameter(τ in Eq 5.3) and set 6 as the meta-learner scaled dimension(l is the scaled dimension of f_γ in Eq 5.4), and set 2 as the loss weight hyperparameter(λ in Eq 5.13). We train our model for 128,000 episodes in eight NVIDIA RTX A5000 GPU(except

for the larger SSv2-all, we train our model for 256,000 episodes). We optimize the MASTAF model with SGD, in which the learning rate is 0.01. After fine-tuning, we adopt the batch-size of 128, 64, 32, 32 for UCF101, HMDB51, SSV2-part, and SSV2-all, respectively.

5.4.2. Comparison with State-of-the-art Algorithms. Table 5.1 tabulates the overall 5-way 1-shot and 5-way 5-shot performance compared with existing methods on two splits of SSv2. We can categorize these comparative methods into three groups based on the embedding network. In 2D CNN embedding group, TSN++ [**103**], CMN-J [**95**], FEAT [**104**] are model agnostic and do not apply any frame-level temporal alignment. Compared with these three methods, the other three methods,i.e., OTAM [**75**], PAL [**81**], and TRX [**79**], adopt frame-level temporal alignment, which further improves the performance of few-shot video classification. MASTAF-{TSN} outperforms existing 5-way 1-shot video classification algorithms in the 2D CNN group. TRX [**79**] achieves SOTA performance for 5-way 5-shot learning because it leverages the temporal information from different frames in different videos in the support set. However, this complex alignment strategy leads to huge computation costs and increases model inference's runtime. Figure 5.4 and Figure 5.5 compare the TFLOPs and model inference's runtime of TRX [**79**] and MASTAF-{TSN}. Our approach achieves SOTA accuracy without increased computational cost and is more efficient than TRX [**79**]. As the number of frames sampled from a video increases, TRX [**79**] consumes more computational resources and takes longer for the inference process. In the 3D CNN group, the accuracy of the TRX-{R3D} is lower than TRX because it cannot perform the frame-level temporal alignment. For PAL [**81**], Zhu [**81**] also mentioned that 3D CNN models [**82**,**84**,**85**] do not perform better than 2D CNN models due to the lacking of frame-level similarity scores. In comparison, MASTAF-{R3D} takes advantage of the spatio-temporal representation from R3D and further improves the performance. In the Transformer group, MASTAF-{ViViT} further enhances the performance. These results demonstrate MASTAF works best when spatio-temporal information is well represented in advanced video classification models. In contrast, existing work with a 2D embedding network cannot maintain high performance when replacing a 2D CNN embedding network with other advanced video representation models.

Table 5.2 tabulates the overall 5-way 1-shot and 5-way 5-shot performance compared with existing methods on UCF101 and HMDB51. Our MASTAF with a 2D embedding network achieves

TABLE 5.1. Comparison on 5-way 1-shot and 5-shot benchmarks of SSv2-part, and SSv2-all. The best performance in each group is highlighted. †: Results from [**75**]. *: Results from [**81**]

Method	Embedding Groups	SSv2-part		SSv2-all	
		1-shot	5-shot	1-shot	5-shot
TSN++† [**103**]	2D CNN	-	-	34.4	43.8
CMN-J [**95**]		36.2	48.8	-	-
FEAT* [**104**]		-	-	45.3	61.2
OTAM [**75**]		-	-	42.8	52.3
PAL [**81**]		-	-	46.4	62.6
TRX [**79**]		36.0	**59.1**	42.0	**64.6**
MASTAF-{TSN}		**37.5**	50.2	**46.9**	62.4
TRX-{R3D}	3D CNN	26.1	47.0	34.9	58.9
MASTAF-{R3D}		**39.9**	**52.2**	**50.3**	**66.7**
TRX-{ViViT}	Transformer	34.7	-	42.7	-
MASTAF-{ViViT}		**45.6**	-	**60.7**	-

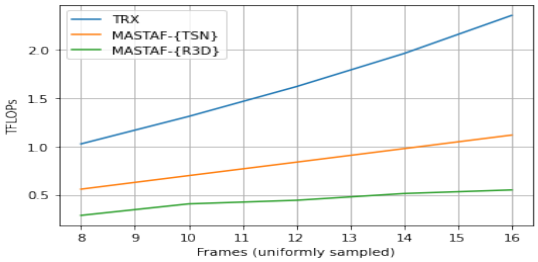


FIGURE 5.4. Computational demand analysis for TRX, MASTAF-{TSN} and MASTAF-{R3D} as the number of sampled frames varies from 8 to 16 frames on UCF101

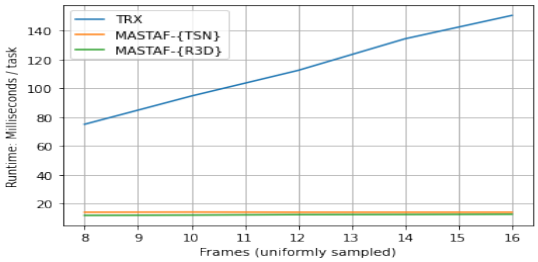


FIGURE 5.5. Model inference's runtime analysis for TRX, MASTAF-{TSN} and MASTAF-{R3D} in one NVIDIA RTX A5000 GPU as the number of sampled frames varies from 8 to 16 frames on UCF1011

decent performance while TRX and PAL achieve SOTA accuracy on these two datasets. The reason is that TSN does not provide enough spatio-temporal information for MASTAF to distinguish the query video from the videos in the support set. So to benefit the most from MASTAF, we explore our MASTAF with a 3D CNN embedding network and video Transformer. As shown in Table 5.2, our MASTAF-{R3D} outperforms other methods based on 3D models and MASTAF-{ViViT} outperforms TRX-{ViViT} and achieves new SOTA performance. Compared with MASTAF-{TSN},

MASTAF-{R3D} has significantly lower resource consumption and running time, as shown in Figure 5.4 and Figure 5.5.

TABLE 5.2. Comparison on 5-way 1-shot and 5-shot benchmarks of UCF101, and HMDB51. The best performance in each group is highlighted. *: Results from [**81**]

Method	Embedding Groups	UCF101		HMDB51	
		1-shot	5-shot	1-shot	5-shot
FEAT* [**104**]	2D CNN	83.9	94.5	60.4	75.2
PAL [**81**]		**85.3**	95.2	**60.9**	**75.8**
TRX [**79**]		-	**96.1**	-	75.6
MASTAF-{TSN}		79.3	90.3	54.8	67.7
ProtoGAN [**97**]	3D CNN	57.8	80.2	34.7	54
ARN [**80**]		66.3	83.1	45.5	60.6
TRX-{R3D}		82.5	94.1	57.0	74.3
MASTAF-{R3D}		**90.6**	**97.6**	**67.9**	**81.2**
TRX-{ViViT}	Transformer	84.8	-	58.1	-
MASTAF-{ViViT}		**91.6**	-	**69.5**	-

5.4.3. Ablation study. We have shown in Section 4.2 that our MASTAF can make the most of the advanced video classification model to improve the accuracy without more computational cost. We now perform detailed ablation studies on two dataset UCF101 and SSV2-all to show each module's influence. In these ablation studies, all MASTAF models use the 3D ResNet-50 model pre-trained on the merged video dataset with Kinetics-700 [**101**],Moment-in-time [**105**], and START-action [**106**] as the embedding network.

TABLE 5.3. Comparison results between the MASTAF without multi-task training setting and MASTAF for 5-way 1-shot video classification

Method	UCF101	SSv2-all
MASTAF-No-Global	89.4	49.5
MASTAF	**90.6**	**50.3**

5.4.3.1. *Multi-task learning setting.* We add a global video classification task in the multi-task learning setting. Table 5.3 shows the comparison results in which we fixed other hyperparameters but without global video classification task in the baseline model(MASTAF-No-Global). From the results, we can see that the global classification task improves the performance, which demonstrates the benefits of the multi-task learning setting. We argue that the global classification task using

TABLE 5.4. Comparison results with three variants of MASTAF for 5-way 1-shot video classification

Method	UCF101	SSv2-all
MASTAF-Neighbor	82.7	43.2
MASTAF-Self	90.3	49.4
MASTAF-Cross	90.5	49.2
MASTAF	**90.6**	**50.3**

the representations from the cross-attention module reduces the risk of overfitting for the nearest neighbor classification task in the training dataset and generates a general representation for unseen class in a few-shot scenario.

5.4.3.2. *Attention fusion mechanism.* To explore the effectiveness of the attention fusion mechanism, we introduce three comparison models, i.e., MASTAF-Neighbor, MASTAF-Self, MASTAF-Cross. In MASTAF-Neighbor, representations learned from the embedding network are fed into the nearest neighbor classifier and a global video classifier directly without our attention mechanisms. For MASTAF-Self and MASTAF-Cross, before being fed into two classifiers, representations go through the self-attention and cross-attention mechanism, respectively. Table 5.4 shows the comparison results. Compared with MASTAF-Neighbor, after adding the attention mechanism, all three other models have a significant performance improvement, demonstrating that representations after the embedding network have some spatio-temporal features related to the non-target action region. The cross-attention mechanism in MASTAF-Cross aid in highlighting the spatio-temporal features associated with the target action region among the query video and support set. MASTAF-Self's self-attention module helps highlight spatio-temporal features related to the action in each video itself. Therefore, combining two different attention modules can take advantage of each module to further extract more discriminative spatio-temporal representations. The results in Table 5.4 demonstrate our argument. We also provide three positive cases to demonstrate the effect of fusion mechanism in the appendix.

5.4.3.3. *Meta-learner.* We evaluate the influence of meat-learner in the MASTAF by developing a model without the meta-learner,i.e., MASTAF-NoML-Mean. In MASTAF-NoML-Mean, we use the average pooling on each relation map(M^{self} in Eq 5.2) and correlation map($M^{cross}_{S_q \leftarrow S^c}$ in Eq 5.6 and $M^{cross}_{S^c \leftarrow S_q}$ in Eq 5.7) as the kernel to compute the attention map in each self-attention module

TABLE 5.5. Comparison results between MASTAF-NoML-Mean and MASTAF for 5-way 1-shot video classification

Method	UCF101	SSv2-all
MASTAF-NoML-Mean	89.9	49.3
MASTAF	**90.6**	**50.3**

TABLE 5.6. Comparison results between MASTAF-NoRes and MASTAF for 5-way 1-shot video classification

Method	UCF101	SSv2-all
MASTAF-NoRes	88.9	49.2
MASTAF	**90.6**	**50.3**

and cross-attention module. As we can see from Table 5.5, our MASTAF with meta-learner outperform MASTAF-NoML-Mean, which means the meta-learner dynamically generates the kernel to summarize the local features in each relation and correlation map.

5.4.3.4. *Residual structure in the attention network.* To verify the effectiveness of residual structure in the attention network, we create a baseline model, i.e., MASTAF-NoRes, in which we remove the residual design in both the self-attention module and cross-attention module. The result in Table 5.6 shows that our MASTAF outperforms the MASTAF-NoRes, which demonstrates the residual structure is beneficial for few-shot video classification because it helps to remain the similar representation for the videos from the same classes and call attention to the minor differences for videos from the different classes.

5.5. Conclusion

This work proposes a Model-Agnostic Spatio-Temporal Attention Fusion network(MASTAF) for few-shot video classification. MASTAF is a simple and effective few-shot video classification framework compatible with different video classification models. MASTAF make the most of the knowledge learned from the advanced video classification model and uses self- and cross-attention to highlight the spatio-temporal features. MASTAF works best when spatio-temporal information is well represented in advanced video classification models and improves the state-of-the-art performance of 5-way 1-shot, and 5-shot video classification on UCF101, HMDB51, and SSv2, e.g.,

MASTAF improves the accuracy of 5-way 1-shot video classification to 91.6%, 69.5%, and 60.7% for UCF101, HMDB51, and SSv2, respectively.

CHAPTER 6

MU-MAE: Multimodal Masked Autoencoders-Based One-Shot Learning

Aely recognizing human activities with multimodal sensors faces challenges due to the labor-intensive nature of data collection and annotation, and reliance on external pretrained models or additional data. To address these challenges, we introduce Multimodal Masked Autoencoders-Based One-Shot Learning (Mu-MAE). Mu-MAE integrates a multimodal masked autoencoder with a synchronized masking strategy tailored for wearable sensors. This masking strategy compels the networks to capture more meaningful spatiotemporal features, which enables effective self-supervised pretraining without the need for external data. Furthermore, Mu-MAE leverages the representation extracted from multimodal masked autoencoders as prior information input to a cross-attention multimodal fusion layer. This fusion layer emphasizes spatiotemporal features requiring attention across different modalities while highlighting differences from other classes, aiding in the classification of various classes in metric-based one-shot learning. Comprehensive evaluations on MMAct one-shot classification show that Mu-MAE outperforms all the evaluated approaches, achieving up to an 80.17% accuracy for five-way one-shot multimodal classification, without the use of additional data.

6.1. Introduction

Human Activity Recognition plays a pivotal role in design and deployment of intelligent systems across various domains, ranging from healthcare and assistive technologies to smart homes and autonomous vehicles [**28**, **109**, **110**, **111**]. For instance, precise activity recognition can facilitate collaborative robots in assisting workers by delivering tools at the right moment [**112**].

In the last decade, extensive research in the field of HAR has been fueled by the spread of smart devices equipped with built-in wearable sensors, high-resolution visual devices, and advancements

in artificial intelligence technology. This research has predominantly centered around the use of unimodal sensor data, such as wearable sensors [**5**,**13**,**113**] and visual inputs [**79**,**86**,**114**]. However, unimodal algorithms encounter difficulties in certain real-world scenarios, especially when it comes to distinguishing similar activities using a single modality, such as differentiating between carrying a light and a heavy object [**115**]. As a response, incorporating additional modalities to support activity identification and improve overall accuracy has become a viable and increasingly popular direction.

Multimodal HAR aims to develop models capable of processing and correlating information from various modalities [**116**]. The use of multimodal representation is anticipated to enhance the performance of human activity recognition, as each modality has the potential to capture distinct and complementary information. Nevertheless, existing multimodal learning approaches face two critical challenges in real-world settings.

Firstly, the collection and annotation of multimodal data is labor-intensive as the number of data modalities increases. Specifically, when transferring a pretrained multimodal model to a target dataset with vision modalities, there is a requirement for a noteworthy amount of annotated multimodal data pertaining to the novel classes in the target dataset for fine-tuning. In the absence of sufficient labeled multimodal data, the performance of multimodal classification is likely to decline.

Secondly, since many multimodal approaches contain large-scale models like ResNet [**70**] and transformers [**99**], especially when dealing with high-dimensional data such as videos, external pretrained models or extra data are necessary for model pretraining [**86**, **117**]. Without external pretrained models or extra data, most multimodal approaches may produce unsatisfactory results, consequently diminishing their applicability in multimodal scenarios.

To overcome these challenges, we present a novel approach named Multimodal Masked Autoencoders-Based One-Shot Learning (Mu-MAE)(as shown in Fig 6.1). Firstly, we introduce one-shot multimodal learning to significantly reduce the annotation cost associated with multimodality data. In one-shot multimodal classification, the multimodal samples in the training and test sets come from different classes, specifically unseen classes in the test set. The objective of a one-shot multimodal classification model is to classify an unlabeled multimodality sample (query) to the

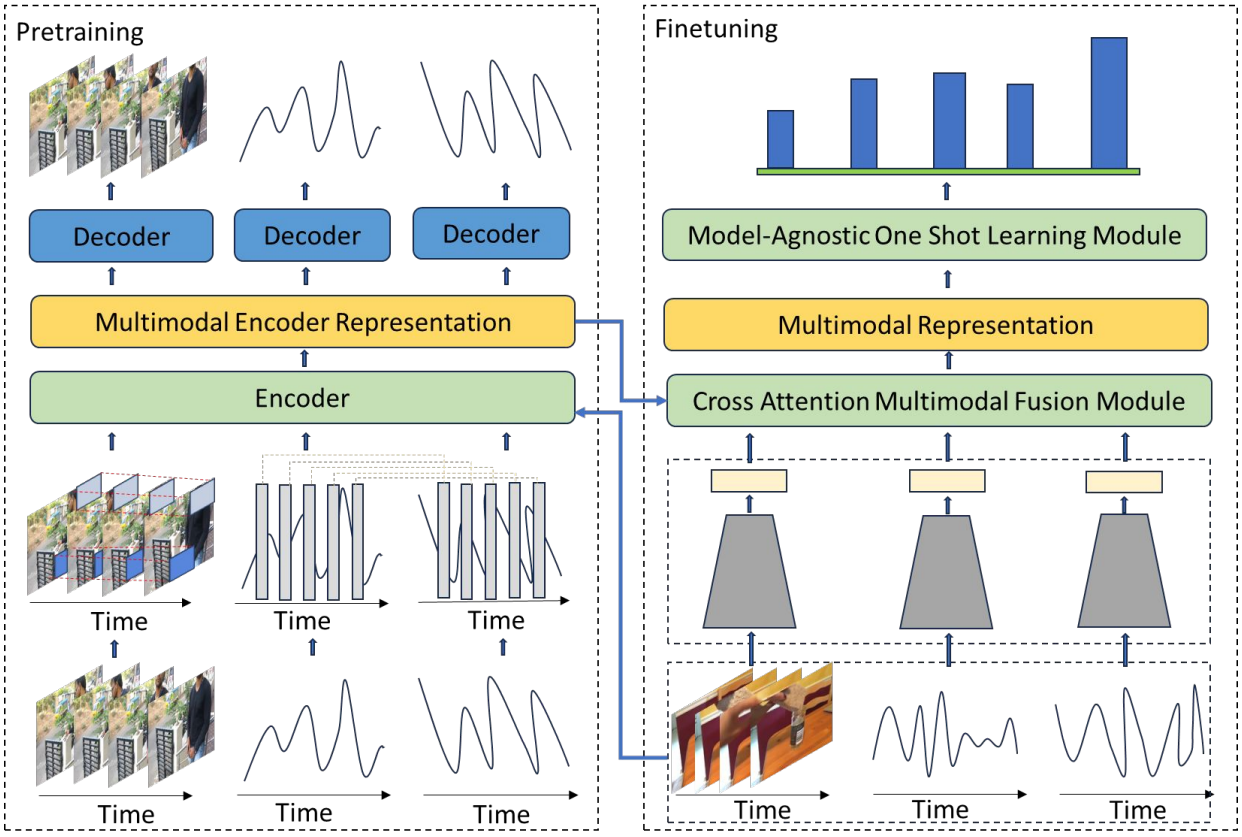


FIGURE 6.1. Illustration of the Multimodal Masked Autoencoders-Based One-Shot Learning (MU-MAE), involving a video modality and two time series modalities. The MU-MAE framework involves two steps. In the first step, known as the pretraining process, a tube masking strategy is employed, and then we get representations of unmasked video data, inspired by the VideoMAE framework [**118**]. Simultaneously, a synchronized masking strategy is applied to the other two physical sensor modalities. This synchronized masking strategy entails masking all time series data at the same specific time points. The concatenated representation, including position information, is then fed into the encoder module to produce the multimodal encoder representation. Subsequently, individual decoders are trained for each modality using mean square error loss to reconstruct the respective modality data. The second step involves a finetuning process focused on one-shot multimodal classification. Unimodal feature encoders pretrained in the pretraining process are applied to extract unimodal representations. The unimodal representations and the multimodal encoder representations are fed into the cross attention multimodal fusion module. This process produces the multimodal representation, which is then directed into the model-agnostic one-shot learning module for classification. More details can be found in Section 6.3.

unseen class (support set). Secondly, for efficient multimodal model pretraining without relying on external data or pretrained models, we propose a multimodal masked autoencoders with a

synchronized masking strategy for wearable sensor data. These masking strategies compel the networks to capture more meaningful spatiotemporal features, thereby making multimodal masked autoencoders a more intricate yet rewarding self-supervised learning task. Lastly, we utilize the multimodal representation extracted from multimodal masked autoencoders as prior information input to the cross-attention multimodal fusion layer. This fusion layer highlights spatiotemporal features that require attention across different modalities while emphasizing differences from other classes. Further details can be found in Section 6.3.

Contributions. We make the following contributions.

1.We present Mu-MAE, an effective and efficient one-shot classification model guided by multimodal masked autoencoders. Mu-MAE possesses the capability to train a vanilla multimodal model directly on the multimodality dataset without relying on any pretrained model or external multimodal data.

2. We design a fusion mechanism that integrates cross-attention networks with the input of multimodal representation learned from the task of reconstructing multimodal data. This integration significantly augments the crucial spatial and temporal regions within the multimodal representation, contributing to the efficacy of the one-shot learning architecture.

3. We conduct a thorough evaluation of Mu-MAE on the one-shot data split of MMAct [**119**], alongside recent multimodal approaches, namely HAMLET [**116**] and MuMu [**115**]. In comparison to these existing works, Mu-MAE enhances state-of-the-art performance without the need for any pretrained model or additional data, achieving 80.17% for five-way one-shot classification. Our code and the one-shot data split of MMAct are available at `https://anonymous.4open.science/r/mu-mae-CAC4`.

6.2. Related work

Multimodal classification. Earlier multimodal learning approaches primarily focused on extracting representations from similar modalities [**120**, **121**, **122**]. For instance, the two-stream CNN excelled at capturing spatial and temporal features from visual data [**121**], while Feichtenhofer's two-stream learning model varied data sampling rates to extract spatial-temporal features [**123**]. Recent

research underscores the development of multimodal learning methods that effectively leverage complementary features from different modalities to overcome dependencies on single-modality data in modality-specific HAR models. For example, in [**124**], they first use attention model to extract unimodal features, which are then fused to generate multimodal representations. Challenges persist in efficiently fusing various unimodal features, leading to the exploration of different fusion approaches, including early fusion, late fusion, and hybrid fusion strategies [**121**,**123**,**125**]. Simonyan et al.'s two-stream CNN architecture [**121**], incorporating spatial and temporal networks, has been extensively studied and proven effective in recent works, employing residual connections [**120**] and slow-fast network techniques [**123**]. Other investigations focus on simultaneous feature fusion from diverse modalities, such as video, and wearable sensor modalities. HAMLET [**116**] employs a hierarchical architecture with a multi-head self-attention mechanism to encode spatio-temporal features from unimodal data in the lower layer and then fuse them in upper layer. MuMu [**115**] incorporates an auxiliary task involving activity group classification to guide the fusion with unimodal representations. Despite these advancements, the ongoing challenge in the field lies in dynamically selecting unimodal features to generate multimodal features. In our Mu-MAE, we address this challenge using a cross-attention multimodal fusion module, dynamically highlighting spatiotemporal features that require attention across different modalities while emphasizing differences from other classes.

Masked visual modeling, Masked visual modeling has proven to be a robust strategy for acquiring impactful representations by employing a sequential process involving masking and subsequent reconstruction. Although early efforts predominantly concentrated on the image domain, employing techniques such as denoised autoencoders [**126**] and convolutions for inpainting missing regions [**127**], recent advancements have expanded the scope of this methodology to encompass videos. Vision transformer architectures like BEiT [**128**], BEVT [**129**], and VIMPAC [**130**] were inspired by language models [**131**,**132**], opting for the prediction of discrete tokens to glean visual representations from both images and videos. The introduction of MAE [**133**] brought forth an asymmetric encoder-decoder architecture finely tuned for masked image modeling, whereas VideoMAE [**118**] took a distinctive approach by directly reconstructing pixels in a more straightforward yet highly effective video masked autoencoder. The evolution in masked visual modeling signals a notable shift towards direct pixel-level reconstruction, enhancing self-supervised pretraining in both

image and video domains. Subsequently, VideoMAE V2 [**134**] introduced an effective pretraining method utilizing a dual masking strategy. In this approach, an encoder processes a subset of video tokens, and a decoder manages another subset of video tokens. This strategy facilitates the efficient pretraining of billion-level models in the video domain.

One-shot learning, One-shot learning algorithms are typically classified into three primary categories: optimization-based methods [**?**,**?**], model-based methods [**91**,**92**], and metric-learning-based methods [**79**, **93**, **114**, **135**]. Among these, metric-learning-based methods emerge as particularly promising, as evidenced by their superior performance in prior studies [**79**, **114**, **135**]. Metric-learning-based approaches compute the distance between representations of support and query samples, utilizing the nearest neighbor for classification. The fundamental principle involves maintaining closeness between representations of similar classes while ensuring differentiation between representations of dissimilar classes. For instance, MASTAF [**114**] highlights spatio-temporal features that demand attention for each class, simultaneously accentuating distinctions from other classes. In our Mu-MAE, we also leverage the metric-learning-based method. To optimize its effectiveness, we employ a cross-attention multimodal fusion module to enhance the differentiation of each class's spatio-temporal features, contributing to improved performance in one-shot classification tasks.

6.3. Proposed Method

6.3.1. Problem definition. A C-way one-shot multimodal learning problem involves learning multimodal representation for model-agnostic one-shot learning, where C denotes the number of categories in the support set. Similar to the one-shot learning problem, we aim to recognize a set of multimodal data into one of given annotated categories, by assessing the similarity between pairs of multimodal representations (R^m) from N heterogeneous modalities, where N denotes the number of modalities.

We use $X^r = \{X_1^r, ..., X_i^r, ..., X_N^r\}$ to denote the raw feature of N heterogeneous modalities and X_i^r stands for raw feature of i modality. The final goal is to get R^m from X^r and then predict R^m to one of the classes.

6.3.2. Approach Overview. Our proposed Multimodal Masked Autoencoders-Based One-Shot Learning consists of four learning modules (as shown in Fig 6.1):

Unimodal Embedding module extracts the representation for each modality.

Multimodal Masked Autoencoders engages in the pretraining of each Unimodal representation encoder, extracting the multimodal representation from both video and physical sensor modalities, which serves as the q value in the fusion layer during the finetuning process.

Cross attention multimodal fusion module integrates representations of all modalities through the utilization of cross-attention mechanisms.

Model-agnostic one-shot learning module classifies a query multimodal instance based on the similarity between the representation of the query and the representation of each class in the support set.

6.3.3. Unimodal Embedding module. In the Mu-MAE model, the goal of the unimodal embedding module is to learn the spatio-temporal representations for each modality. We use $f_m^{\varphi} = \{f_1^{\varphi}, ..., f_i^{\varphi}, ..., f_N^{\varphi}\}$ to denote the unimodal embedding module of N heterogeneous modalities and f_i^{φ} stands for unimodal embedding module of i modality.

Given a raw feature extracted from the i modality, X_i^r, let U_i denote the representation learned from the unimodal embedding module:

$$U_i = f_i^{\varphi}(X_i^r). \tag{6.1}$$

6.3.4. Multimodal Masked Autoencoders. The multimodal masked autoencoders module has two primary objectives. Firstly, it aims to train a vanilla unimodal embedding network for each modality directly on the multimodal dataset, without any pretrained models. Secondly, the multimodal representations extracted from the reconstruction task serve as prior information for the efficient fusion of multimodal representations in one-shot classification tasks.

As shown in Fig 6.1, we first adopt a tube masking strategy and then get the representation of unmasked video data R_{unmask}^{v}, inspired by the approach employed in the VideoMAE framework [**118**]. Then for other physical sensor modalities, we utilize the synchronized masking strategy to get the representations for each unmasked sensor data. We use $R_{unmask}^{s} = [R_{unmask}^{s,1}; ...' R_{unmask}^{s,i}; ..., R_{unmask}^{s,N-1}]$ to denote the concatenated representation of all the unmasked sensor data, which

$R^{s,i}_{unmask}$ denotes the representation of i unmasked sensor data. The synchronized masking strategy involves simultaneously masking all time series data at the same specific time points. This approach is helpful for mitigating information leakage during masked modeling and make masked time series data reconstruction a meaningful self-supervised pretraining task. Then we get the concatenated representation with position information of all the unmasked modality data as:

$$R^{m}_{unmask} = [R^{v}_{unmask} + P^{v}; R^{s}_{unmask} + P^{s}], \tag{6.2}$$

which P^v and P^s denote the position information of video and other sensor modalities.

Multimodal Masked Autoencoders Encoder is based on the ViT architecture [**107**], exclusively applied to visible, unmasked patches, inspired by the approach employed in the VideoMAE framework [**118**]. We use $f_{encoder}$ to denote the multimodal masked autoencoder encoder. Then we compute the representation $R^{m}_{encoder}$ extracted from encoder as:

$$R^{m}_{encoder} = f_{encoder}(R^{m}_{unmask}). \tag{6.3}$$

Utilizing a masking strategy provides the benefit of training large encoders while requiring less computational resources and memory.

Multimodal Masked Autoencoders Decoder is designed in a light weight setting following the design in the video domain [**118**], which could significantly reduces pretraining time. We use $f_{decoder}$ to denote the multimodal masked autoencoder decoder. Then we compute the representation $R^{m}_{decoder}$ extracted from encoder as:

$$R^{m}_{decoder} = f_{decoder}(R^{m}_{encoder}). \tag{6.4}$$

After that, we employ the Mean Squared Error (MSE) loss as the loss function for tasks related to the reconstruction of multimodal data.

6.3.5. Cross attention multimodal fusion module. We use the representations from multimodal masked autoencoders encoder as prior information, $R^{m}_{encoder}$, to extract multimodal representations. One benefit is to highlight spatio-temporal features that need attention across different modalities while increasing the differences from other classes. To compute the attended multimodal

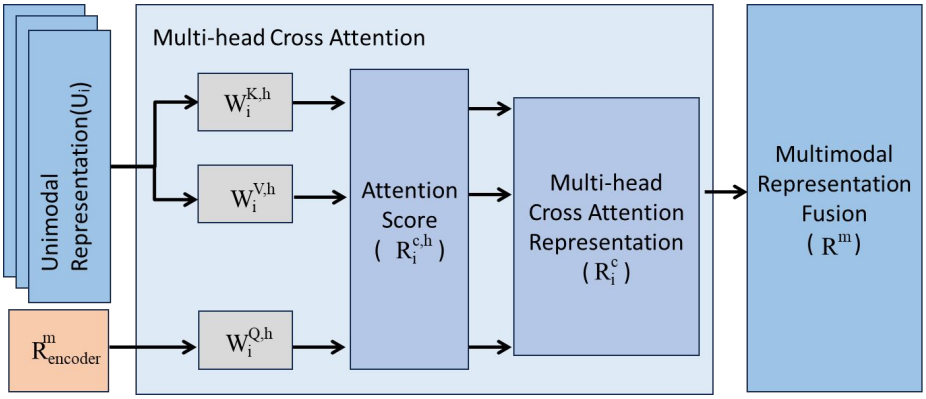


FIGURE 6.2. Cross Attention Multimodal Fusion Module. $R^m_{encoder}$ is the multimodal representation from multimodal masked autoencoders' encoder.

representation, we utilize multi-head cross attention method (as shown in Fig 6.2). First, we transforms the extracted unimodal features of the i modality, U_i ,to generate unimodal key ($K_i^{u,h}$) and value ($V_i^{u,h}$) feature vectors for head h using the following procedure:

$$K_i^{u,h} = U_i W_i^{K,h}; V_i^{u,h} = U_i W_i^{V,h}, \tag{6.5}$$

which $W_i^{K,h}$ and $W_i^{V,h}$ are learnable parameters. Then, we transforms the extracted representations from multimodal masked autoencoders' encoder, $R^m_{encoder}$, to generate the query feature vectors $Q_i^{u,h}$ as:

$$Q_i^{u,h} = R^m_{encoder} W_i^{Q,h}, \tag{6.6}$$

which $W_i^{Q,h}$ is a learnable parameter. Then we use query feature vectors $Q_i^{u,h}$ of the i modality for head h to generate the multimodal representation of the i modality for head $h$as:

$$R_i^{c,h} = softmax(\frac{Q_i^{u,h}(K_i^{u,h})^T}{exp(d_i^K)} V_i^{u,h}), \tag{6.7}$$

which d_i^K is dimension of $K_i^{u,h}$. After that, all the head multimodal representations of the i modality $R_i^{c,h}$ are concatenated and projected to produce multi-head cross attention representation of the i modality (R_i^c).

$$R_i^c = [R_i^{c,1} : ... : R_i^{c,h}] W_i^c, \tag{6.8}$$

where W_i^c is the projection parameter. Following that, we concatenate the multi-head cross attention representations from all modalities and employ a linear projection to generate the multimodal representation R^m.

$$R^m = W^m[R_1^c : ...R_i^c : ... : R_N^c], \tag{6.9}$$

which W^m is a learnable projection parameter.

6.3.6. Model-agnostic one-shot learning module. Once we obtain R^m, including spatio-temporal features with knowledge across different modalities, we apply the existing model-agnostic one-shot learning module for one-shot learning task. The model-agnostic one-shot learning module assesses the distance between the representations of support samples and query sample, and classifies them with the aid of the nearest neighbor to keep similar classes close and dissimilar classes far away. We represent the R^m of support class k as $R_{s_k}^m$ and the R^m of a query sample as R_q^m. The model-agnostic one-shot learning module is denoted by $f_{one-shot}$. Then we compute the one-shot learning representation of support class k ($S_{s_k}^m$) and query sample (S_q^m) as:

$$S_{s_k}^m = f_{one-shot}(R_{s_k}^m), \tag{6.10}$$

$$S_q^m = f_{one-shot}(R_q^m). \tag{6.11}$$

After that, we compute the probability of predicting S_q^m as the class k using one-shot learning representation:

$$P(y = k|S_q^m) = \frac{exp(-D_{cos}(S_q^m, S_{s_k}^m))}{\sum_{j=1}^{C} exp(-D_{cos}(S_q^m, S_{s_j}^m))}, \tag{6.12}$$

where D_{cos} denotes the cosine distance and $P(y = k|S_q^m)$ denotes the probability of predicting S_q^m as the class $k \in \{1, 2, ..., C\}$ using one-shot learning representations. C represents the number of categories in the support set.

Method	Backbone	Extra data	Fusion type	Accuracy(SD)
HAMLET [**116**]	ResNet50	no external data	sum	41.77(+/-0.97)
	ResNet50	no external data	concat	42.18(+/-0.97)
	ResNet50	ImageNet-1k	sum	75.87%(+/-0.81)
	ResNet50	ImageNet-1k	concat	76.05%(+/-0.83)
	ViT	no external data	concat	40.02(+/-0.92)
	ViT	Kinetics-400	concat	78.34%(+/-0.79)
MuMu [**115**]	ResNet50	no external data	concat	46.02%(+/-0.98)
	ResNet50	ImageNet-1k	concat	78.65%(+/-0.80)
	ViT	no external data	concat	45.16%(+/-0.94)
	ViT	Kinetics-400	concat	80.22%(+/-0.84)
Mu-MAE	ViT	no external data	concat	**80.17%(+/-0.78)**
Mu-MAE	ViT	Kinetics-400	concat	**83.82%(+/-0.77)**

TABLE 6.1. Comparison on 5-way 1-shot benchmarks of MMAct. SD stands for standard deviation. The best performances with or without extra data are highlighted.

Finally, we use a negative log-probability as the loss function of the nearest neighbor classifier based on the one-shot class label:

$$L = -\sum_{k=1}^{C} logP(y = k|S_q^m). \tag{6.13}$$

6.4. Evaluation

6.4.1. Experimental Setup. Datasets. There are no established one-shot data splits available for one-shot multimodal classification involving both video and wearable sensors. Thus, we undertake the random division of classes into meta-training/validation sets and a meta-testing set within the MMAct [**119**] for the few-shot multimodal classification evaluation. This data split is presented at (`https://anonymous.4open.science/r/mu-mae-CAC4`) for future research.

After eliminating classes lacking data from all five modalities (video, accelerometer from phone, accelerometer from watch, gyroscope, and orientation), the MMAct dataset comprises 33 activities [**119**], with an average of over 1,000 data samples for each activity across all five modalities. These 33 activities are then split into non-overlapping sets, with 23 assigned for use as the meta-training/validation set and 10 designated for the meta-testing set.

Experimental Configuration. Following the evaluation process in state-of-the-art one-shot learning algorithms [**79**, **103**, **114**], we evaluate the 5-way 1-shot multimodal classification task and

report the average accuracy over 10,000 randomly selected episodes from the test set. We compare our results with two state-of-the-art multimodal classification algorithms, i.e., HAMLET [**116**], MuMu [**115**]. In particular, MuMu necessitates the categorization of all training classes into three groups, i.e., complex, simple, desk [**115**]. To accommodate this requirement, we group the 23 activities into these three distinct categories. We use ViT-B [**107**] as our video embedding network, and 1-D CNN as the unimodal embedding network for data from other sensor modalities. We use MASTAF [**114**] as our model-agnostic one-shot learning module, as it has demonstrated state-of-the-art performance with the model-agnostic embedding. The unimodal features from physical sensor modalities are encoded into 64-sized feature embeddings. During the pretraining phase of the multimodal masked autoencoders module, we extract 16 frames from each video. In the subsequent finetuning process of MASTAF [**114**], 8 frames are utilized. The pretraining of the multimodal masked autoencoders is conducted on 8 NVIDIA RTX A5000 GPUs, spanning 800 epochs, while the finetuning experiments for one-shot learning involve 256,000 episodes. PyTorch and DeepSpeed [**136**] frameworks are utilized for expedited pretraining, and finetuning is carried out using Stochastic Gradient Descent.

6.4.2. Comparison with State-of-the-art Algorithms. Table 6.1 presents a comprehensive comparison of the overall 5-way 1-shot performance against existing methods on the MMAct one-shot data split. MASTAF serves as the chosen one-shot learning module across all algorithms, and the reported average accuracy is based on 10,000 randomly selected episodes from the test set. For the HAMLET method, we explore two fusion merge types: concatenation-based fusion and summation-based fusion. Both fusion methods are re-implemented for comparative analysis. Additionally, pretrained and trained-from-scratch ResNet 50 [**70**] and ViT [**86**] are included as embedding networks for both HAMLET and MuMu. In our method, Mu-MAE, we conduct the vanilla ViT [**86**] and pretained ViT [**118**] on the Kinetics-400 as the video embeding network with varying mask ratios and decoder depths in the pretraining process, reporting the best performance achieved with an 85% mask ratio and 4 blocks of the decoder. As shown in Table 6.1, Mu-MAE without any external data outperforms trained-from-scratch state-of-the-art methods, namely HAMLET [**116**] and MuMu [**115**], demonstrating improvements of 40.15% and 35.01% in average accuracy, respectively. The lower performance of trained-from-scratch HAMLET [**116**]

Method	Accuracy
Mu-MAE-scratch	41.21%(+/-0.91)
Mu-MAE-without-cross	78.46%(+/-0.81)
Mu-MAE	**80.17%(+/-0.78)**

TABLE 6.2. Comparison results with two variants Mu-MAE for 5-way 1-shot classification on MMAct. The best performance is highlighted.

blocks	Accuracy	GPU mem.
1	79.52%(+/-0.82)	10.4G
2	79.64%(+/-0.81)	13.1G
4	**80.17%(+/-0.78)**	17.6G

TABLE 6.3. Decoder depth. "GPU mem." is GPU memory during pretraining. The best performance is highlighted.

and MuMu [**115**] is attributed to their inability to leverage ViT [**86**] model scale without any external data. In contrast, Mu-MAE, equipped with a multimodal masked autoencoder, effortlessly scales up with potent backbones (e.g., ViT [**86**]), attaining an accuracy of 80.17% on MMAct [**119**] without relying on external data.

Additionally, in the one-shot multimodal learning task, Mu-MAE with externald data outperforms the pretrained HAMLET and MuMu with an imporvement of 5.48% and 3.6%, reprectively. These improvement highlight Mu-MAE's capacity to generate spatiotemporal features that demand attention across diverse modalities, simultaneously amplifying the distinctions from other classes in one-shot learning.

While Mu-MAE with pretrained ViT [**118**] exhibits superior performance compared to Mu-MAE with vanilla ViT, the slight gap in performance between these two models highlights the efficacy of our multimodal masked autoencoders in achieving good performance using only the target dataset without relying on external data. Our approach reduces computation costs and resource requirements.

6.4.3. Ablation study. As demonstrated in Section 6.4.2, our Mu-MAE shows better performance compared to other state-of-the-art multimodal classification algorithms in a one-shot scenario, even without any external data. We conduct in-depth ablation studies on MMAct to show the impact of each module.

Mask type	Mask ratio	Accuracy
random	85%	79.01%(+/-0.82)
synchronized	75%	79.12%(+/-0.77)
synchronized	85%	**80.17%(+/-0.78)**
synchronized	95%	78.25%(+/-0.81)

TABLE 6.4. Mask design. Comparison results with different mask types and mask ratios for 5-way 1-shot classification on MMAct. The best performance is highlighted.

Multimodal masked autoencoders and Cross attention multimodal fusion. To explore the effectiveness of the multimodal masked autoencoders, we introduce two comparative models, namely Mu-Mae-scratch and Mu-Mae-without-cross. In Mu-Mae-scratch, video representations obtained from the vanilla ViT [**107**] and other physical sensor representations are concatenated and processed through a linear projection to generate the multimodal representation. Subsequently, this multimodal representation is input into the MASTAF [**114**] one-shot learning architecture. Mu-Mae-without-cross is essentially the same as Mu-Mae-scratch, except for the fact that the unimodal embedding networks undergo pretraining using multimodal masked autoencoders.The comparative results are presented in Table 6.2.

In contrast to Mu-MAE-without-cross, the inclusion of the cross-attention multimodal fusion mechanism in Mu-MAE leads to 1.71% performance improvement. This suggests that representations obtained from multimodal masked autoencoders contain valuable spatiotemporal knowledge across various modalities. Consequently, this enriched information facilitates the fusion layer in generating more distinctive features, thereby improving differentiation from other classes. When compared to Mu-MAE-scratch, after adding the multimodal masked autoencoder pretraining process, the other two models achieve significant performance enhancements. It demonstrates that the pretraining of multimodal masked autoencoders is essential for realizing the advantages of the model scale, especially in larger-scale models such as ViT [**86**] and ResNet 50 [**70**].

Decoder design. In our Mu-MAE design, we adopt a lightweight decoder inspired by VideoMAE [**118**]. One advantage of employing a shallow decoder is the reduction in GPU memory consumption, which is particularly beneficial when processing video data. Our experiments involve

variations in decoder depth, as detailed in Table 6.3. We can see from the Table 6.3 that 4 blocks of decoder achieve the best tradeoff.

Masking strategy and mask ratio. In our Mu-MAE design, we implement a synchronized masking strategy for all time series sensor data. One advantage of employing a synchronized masking strategy is to prevent information leakage between these time series sensor data. To evaluate the effect of the synchronized masking strategy, we conduct an experiment with plain random masking, setting the mask ratio at 85%, as outlined in Table 6.4. The results reveal that Mu-MAE with the synchronized masking strategy outperforms its counterpart with plain random masking, achieving a 1.16% improvement in accuracy. When increasing the masking ratio from 75% to 85% for synchronized masking, the performance on 5-way 1-shot multimodal classification boosts from 79.12% to 80.17%. However, when the masking ratio is further increased from 85% to 95% for synchronized masking, the performance on 5-way 1-shot multimodal classification decreases from 80.17% to 78.25%, which means a 85% mask ratio is a good tradeoff for 5-way 1-shot multimodal classification on MMAct [**119**]. These outcomes demonstrate that our synchronized masking designs make the networks to capture more useful spatiotemporal features, making Mu-MAE a more challenging yet rewarding self-supervised learning task.

6.5. Conclusion

This paper proposes a Multimodal Masked Autoencoders-Based One-Shot Learning (Mu-MAE). Mu-MAE is a simple and efficient one-shot multimodal classification framework without using any extra data for pretraining. Mu-MAE makes the most of the knowledge learned from multimodal masked autoencoders and uses a cross-attention multimodal fusion module to highlight the spatiotemporal features for one-shot multimodal classification. Mu-MAE outperforms existing methods (HAMLET and MuMu) by achieving 80.17% for five-way one-shot multimodal classification, without relying on pretrained models or additional data.

CHAPTER 7

Future Directions and Conclusion

7.1. Conclusion

In conclusion, our dissertation explores the burgeoning field of HAR within healthcare, leveraging advancements in sensor technology, particularly focusing on wearable sensors and video sensors. The integration of sensors into healthcare systems offers unprecedented opportunities for real-time monitoring and intervention, benefiting patients with diverse conditions ranging from physical impairments to neurodegenerative diseases.

We begin by highlighting the evolution of sensor technology, emphasizing its impact on the feasibility and efficacy of HAR systems in healthcare settings. We discuss the rising popularity of video surveillance as a cost-effective and efficient solution, particularly in the context of HAR for healthcare applications.

Moreover, we delineate the role of HAR in healthcare systems, emphasizing its significance in tracking patient activities, providing real-time feedback, and facilitating remote monitoring by healthcare professionals. By utilizing wearable sensors and video sensors, HAR systems contribute to personalized patient care, particularly for individuals with chronic conditions like obesity, diabetes, cardiovascular diseases, and neurodegenerative disorders.

We then delve into the methodological aspects of HAR, particularly focusing on machine learning approaches for activity recognition. We discuss the limitations of traditional feature engineering techniques and underscore the potential of deep learning methods in capturing complex activity patterns from sensor data. Despite challenges such as data sparsity and annotation efforts, deep learning holds promise for enhancing the performance of HAR systems in healthcare.

Furthermore, we present several novel contributions in the realm of healthcare-oriented HAR systems. These include the development of accurate AI models for Early Mobility Activity (EMA) recognition in Intensive Care Unit (ICU) patients, the design of innovative systems like BWCNN for

communication assistance in neurodegenerative diseases, and the proposal of advanced models such as MASTAF and Mu-MAE to address challenges in few-shot video classification and multimodal sensor fusion, respectively.

Overall, our dissertation offers a comprehensive overview of the current landscape of HAR in healthcare, highlighting key challenges, technological advancements, and innovative solutions. By bridging the gap between medical and computer science disciplines, we open avenues for future research and development aimed at improving patient outcomes through intelligent activity recognition systems.

7.2. Future Work

One promising avenue for future research lies in the integration of contrastive learning techniques into HAR for healthcare applications. In recent years, deep learning-based methods have been widely used in wearable sensor-based activity recognition tasks. Under supervised learning tasks, models such as LSTM [**137**], CNN [**50**], DeepConvLSTM [**138**], DeepConvLSTMAttention [**139**], and Multi-Head Convolutional Attention [**140**] have been proposed to improve the accuracy of HAR significantly. However, this approach usually requires a large number of labeled samples to train a deep learning model, which generally requires manual labeling of sensor data through a time-consuming and tedious process. In healthcare, it is also challenging and expensive to collect a large number of labeled data. In addition, the labeling is affected by various noise sources, such as sensor noise, segmentation problems, and changes in the activities of different people, which make the annotation process error-prone [**57**]. Therefore, the limitation of sensor data annotation is a significant challenge for HAR in healthcare.

Contrastive learning, a dominant form of self-supervised learning in various domains, offers a potential solution to the limitations of data annotation [**141**]. By generating pseudo-labels through data augmentation and training the model to distinguish between positive and negative pairs, contrastive learning enables effective representation learning with minimal labeled data [**142**]. This approach is particularly suitable for healthcare applications where labeled data is scarce and prone to noise.

However, current sensor data augmentation methods in contrastive learning often fall short of outperforming supervised learning approaches, even with limited labeled data [**143**]. Therefore, there is a pressing need to develop novel data augmentation techniques tailored specifically for sensor data in HAR for healthcare. By leveraging the unique features of contrastive learning and optimizing data augmentation strategies, future research can unlock the full potential of self-supervised learning in improving activity recognition accuracy in healthcare settings.

Bibliography

[1] Y. Liu, L. Nie, L. Liu, and D. Rosenblum, "From action to activity: Sensor-based activity recognition," *Neurocomputing*, vol. 181, 11 2015.

[2] A. B. Sargana, P. Angelov, and Z. Habib, *Vision Based Human Activity Recognition: A Review*, vol. 513, pp. 341–371. 01 2017.

[3] L. M. Dang, S. Hassan, S. Im, and H. Moon, "Face image manipulation detection based on a convolutional neural network," *Expert Systems with Applications*, vol. 129, 04 2019.

[4] G. Plasqui, "Smart approaches for assessing free-living energy expenditure following identification of types of physical activity: Assessing free-living energy expenditure," *Obesity Reviews*, vol. 18, pp. 50–55, 02 2017.

[5] R. Varatharajan and G. Manogaran, "Wearable sensor devices for early detection of alzheimer disease using dynamic time warping algorithm," *Cluster Computing*, vol. 21, 03 2018.

[6] O. Lara and M. Labrador, "A survey on human activity recognition using wearable sensors," *Communications Surveys & Tutorials, IEEE*, vol. 15, pp. 1192–1209, 2013.

[7] T. Huynh and B. Schiele, "Analyzing features for activity recognition," in *Proceedings of the 2005 Joint Conference on Smart Objects and Ambient Intelligence: Innovative Context-Aware Services: Usages and Technologies*, sOc-EUSAI '05, (New York, NY, USA), p. 159–163, Association for Computing Machinery, 2005.

[8] E. Brophy, J. J. D. Veiga, Z. Wang, A. F. Smeaton, and T. E. Ward, "An interpretable machine vision approach to human activity recognition using photoplethysmograph sensor data," *ArXiv*, vol. abs/1812.00668, 2018.

[9] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, DMKD '03, (New York, NY, USA), p. 2–11, Association for Computing Machinery, 2003.

[10] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Comput. Surv.*, vol. 51, sep 2018.

[11] S. Ha, J.-M. Yun, and S. Choi, "Multi-modal convolutional neural networks for activity recognition," in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 3017–3022, 2015.

[12] L. M. Dang, K. Min, H. Wang, M. J. Piran, C. H. Lee, and H. Moon, "Sensor-based and vision-based human activity recognition: a comprehensive survey," *Pattern Recognition*, vol. 108, pp. 1–41, 2020.

[13] R. Liu, S. A. Fazio, H. Zhang, A. A. Ramli, X. Liu, and J. Y. Adams, "Early mobility recognition for intensive care unit patients using accelerometers," in *KDD Workshop on Artificial Intelligence of Things (AIoT)*, pp. 1–6, 2021.

[14] A. A. Ramli, H. Zhang, J. Hou, R. Liu, X. Liu, A. Nicorici, D. Aranki, C. Owens, P. Prasad, C. McDonald, and E. Henricson, "Gait characterization in duchenne muscular dystrophy (dmd) using a single-sensor accelerometer: Classical machine learning and deep learning approaches," 2021.

[15] M. H. M. Noor, Z. Salcic, and K. I.-K. Wang, "Adaptive sliding window segmentation for physical activity recognition using a single tri-axial accelerometer," *Pervasive and Mobile Computing*, vol. 38, no. 1, pp. 41–59, 2016.

[16] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Science and Information Conference*, pp. 372–378, 2014.

[17] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: a survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[18] A. C. Castro-Avila, P. Seron, E. Fang, M. Gaete, and S. Mickan, "Effect of early rehabilitation during intensive care unit stay on functional status: systematic review and meta-analysis," *PloS One*, vol. 10, no. 7, pp. 1–21, 2015.

[19] J. Adler and D. Malone, "Early mobilization in the intensive care unit: a systematic review," *Cardiopulmonary Physical Therapy Journal*, vol. 23, pp. 5–13, 2012.

[20] E. M. Yiu and A. J. Kornberg, "Duchenne muscular dystrophy," *Journal of Paediatrics and Child Health*, pp. 759–764, 2015.

[21] Physiopedia, "North start ambulatory assessment." `https://www.physio-pedia.com/North_Star_Ambulatory_Assessment`. [accessed on 29-June-2021].

[22] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8827–8836, 2018.

[23] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. arXiv:1406.2661.

[24] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321–357, 2002.

[25] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher, "Activity recognition and monitoring using multiple sensors on different body positions," in *International Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, pp. 1–4, 2006.

[26] K. V. Laerhoven, A. Schmidt, and H.-W. Gellersen, "Multi-sensor context aware clothing," in *International Symposium on Wearable Computers*, pp. 1–8, 2002.

[27] D. Jarchi, J. Pope, T. K. Lee, L. Tamjidi, A. Mirzaei, and S. Sanei, "A review on accelerometry-based gait analysis and emerging clinical applications," *IEEE Reviews in Biomedical Engineering*, vol. 11, pp. 177–194, 2018.

[28] A. A. Ramli, R. Liu, R. Krishnamoorthy, I. B. Vishal, X. Wang, I. Tagkopoulos, and X. Liu, "Bwcnn: Blink to word, a real-time convolutional neural network approach," in *Internet of Things - ICIOT 2020*, (Cham), pp. 133–140, Springer International Publishing, 2020.

[29] K. Keenan, P. Lovoi, and W. Smith, "The neurological examination improves cranial accelerometry large vessel occlusion prediction accuracy," *Neurocritical Care*, pp. 1–10, 2020.

[30] A. Wijekoon and N. Wiratunga, "Learning-to-learn personalised human activity recognition models," 2020.

[31] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, pp. 1126–1135, 2017.

[32] S. Feng and M. Duarte, "Few-shot learning-based human activity recognition," *Expert Systems with Applications*, vol. 138, pp. 1–12, 2019.

[33] E. Kim, "Interpretable and accurate convolutional neural networks for human activity recognition," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 11, pp. 7190–7198, 2020.

[34] K. Chen, L. Yao, X. Wang, D. Zhang, T. Gu, Z. Yu, and Z. Yang, "Interpretable parallel recurrent neural networks with convolutional attentions for multi-modality activity modeling," 2018.

[35] Y. Iwasawa, K. Nakayama, I. Yairi, and Y. Matsuo, "Privacy issues regarding the application of dnns to activity-recognition using wearables and its countermeasures by use of adversarial training," in *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1930–1936, 2017.

[36] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1–10, 2017.

[37] A. Castro, P. Seron, E. Fan, M. Gaete, and S. Mickan, "Effect of early rehabilitation during intensive care unit stay on functional status: Systematic review and meta-analysis," *PloS one*, vol. 10, p. e0130722, 07 2015.

[38] J. Adler and D. Malone, "Early mobilization in the intensive care unit: A systematic review," *Cardiopulmonary physical therapy journal*, vol. 23, pp. 5–13, 2012.

[39] S. Fazio, A. Doroy, N. Marto, S. Taylor, N. Anderson, H. Young, and J. Adams, "Quantifying mobility in the icu: Comparison of electronic health record documentation and accelerometer-based sensors to clinician-annotated video," *Critical Care Explorations*, vol. 2, p. e0091, 04 2020.

[40] J. Goschenhofer, F. Pfister, K. Yuksel, B. Bischl, U. Fietzek, and J. Thomas, *Wearable-Based Parkinson's Disease Severity Monitoring Using Deep Learning*, pp. 400–415. 2020.

[41] A. Palaniappan, R. Bhargavi, and V. Vaidehi, "Abnormal human activity recognition using svm based approach," pp. 97–102, 2012.

[42] G. Bhat, R. Deb, V. Chaurasia, H. Shill, and U. Ogras, "Online human activity recognition using low-power wearable devices," 2018.

[43] A. Davide, G. Alessandro, O. Luca, P. Xavier, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *ESANN*, 2013.

[44] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. del R. Millán, and D. Roggen, "The opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 2033 – 2042, 2013. Smart Approaches for Human Action Recognition.

[45] D. S. Baim, W. S. Colucci, E. S. Monrad, H. S. Smith, R. F. Wright, A. Lanoue, D. F. Gauthier, B. J. Ransil, W. Grossman, and E. Braunwald, "Survival of patients with severe congestive heart failure treated with oral milrinone," *Journal of the American College of Cardiology*, vol. 7, no. 3, pp. 661 – 670, 1986.

[46] S. Nusser, S. Intille, and R. Maitra, "Emerging technologies and next-generation intensive longitudinal data collection," *Models for Intensive Longitudinal Data*, pp. 254–277, 2006.

[47] B. Frénay and M. Verleysen, "Classification in the presence of label noise: A survey," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 25, pp. 845–869, 2014.

[48] F. Lin, W. Zhongmin, and W. Hai, "Human activity recognition model based on decision tree," in *2013 International Conference on Advanced Cloud and Big Data*, pp. 64–68, 2013.

[49] Z. Zaki, M. A. Shah, K. Wakil, and F. Sher, "Logistic regression based human activities recognition," *Journal of Mechanics of Continua and Mathematical Sciences*, vol. 15, pp. 228–246, 2020.

[50] H. Cho and S. Yoon, "Divide and conquer-based 1d cnn human activity recognition using test data sharpening," *Sensors (Basel, Switzerland)*, vol. 18, 2018.

[51] S. Patel, K. Lorincz, R. Hughes, N. Huggins, J. Growdon, D. Standaert, Y. Akay, J. Dy, M. Welsh, and P. Bonato, "Monitoring motor fluctuations in patients with parkinson's disease using wearable sensors," *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 13, pp. 864–73, 2009.

[52] E. Kańtoch, "Human activity recognition for physical rehabilitation using wearable sensors fusion and artificial neural networks," in *2017 Computing in Cardiology (CinC)*, pp. 1–4, 2017.

[53] J. Hou, Q. Wang, K. Alshebli, L. Ball, S. Birge, M. Caccamo, C.-F. Cheah, E. Gilbert, C. Gunter, E. Gunter, C.-G. Lee, K. Karahalios, M.-Y. Nam, N. Nitya, R. Chaudhri, L. Sha, W. Shin, S. Yu, Y. Yu, and C. Zeng, "Pas: A wireless-enabled, sensor-integrated personal assistance system for independent and assisted living," *Joint Workshop on High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability*, vol. 0, pp. 64–75, 06 2007.

[54] S. Yeung, F. Rinaldo, J. Jopling, B. Liu, R. Mehra, N. Downing, M. Guo, G. Bianconi, A. Alahi, J. Lee, B. Campbell, K. Deru, W. Beninati, L. Fei-Fei, and A. Milstein, "A computer vision system for deep learning-based detection of patient mobilization activities in the icu," *Nature Partner Journals Digital Medicine*, vol. 2, p. 11, 2019.

[55] R. Mehra, G. Bianconi, S. Yeung, and L. Fei-Fei, "Depth-based activity recognition in icus using convolutional and recurrent neural networks," 2017.

[56] P. Morris, A. Goad, C. Thompson, K. Taylor, B. Harry, L. Griffin, A. Ross, L. Anderson, S. Baker, M. Sanchez, L. Penley, A. Howard, L. Dixon, S. Leach, R. Small, R. Hite, and E. Haponik, "Early intensive care unit mobility therapy in the treatment of acute respiratory failure," *Critical care medicine*, vol. 36, pp. 2238–43, 2008.

[57] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, "Deep learning for sensor-based human activity recognition: Overview, challenges and opportunities," 2020.

[58] V. Raj.S, "Pervasive and mobile computing based human activity recognition system," *International Journal of Engineering Research & Technology*, vol. 1, no. 6, pp. 757–760, 2013.

[59] A. Akbari, J. Wu, R. Grimsley, and R. Jafari, "Hierarchical signal segmentation and classification for accurate activity recognition," pp. 1596–1605, 2018.

[60] A. Bayat, M. Pomplun, and D. Tran, "A study on human activity recognition using accelerometer data from smartphones," *Procedia Computer Science*, vol. 34, pp. 450–457, 2014.

[61] W. Jian and R. Jafari, "Orientation independent activity/gesture recognition using wearable motion sensors," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1427–1437, 2019.

[62] H. Singh and J. Singh, "Real-time eye blink and wink detection for object selection in hci systems," *Journal on Multimodal User Interfaces*, vol. 12, 01 2018.

[63] A. Dementyev and C. Holz, "Dualblink: A wearable device to continuously detect, track, and actuate blinking for alleviating dry eyes and computer vision syndrome," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, mar 2017.

[64] A. P. A. A. Mohammed and M. S. S. A. Anwer, "Efficient eye blink detection method for disabled-helping domain," *International Journal of Advanced Computer Science and Applications*, vol. 5, no. 5, 2014.

[65] K. Mukherjee and D. Chatterjee, "Augmentative and alternative communication device based on eye-blink detection and conversion to morse-code to aid paralyzed individuals," in *2015 International Conference on Communication, Information Computing Technology (ICCICT)*, pp. 1–5, 2015.

[66] K. Grauman, M. Betke, J. Gips, and G. Bradski, "Communication via eye blinks - detection and duration analysis in real time," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, pp. I–I, 2001.

[67] M. Su, C. Yeh, S. Lin, P. Wang, and S. Hou, "An implementation of an eye-blink-based communication aid for people with severe disabilities," in *2008 International Conference on Audio, Language and Image Processing*, pp. 351–356, 2008.

[68] A. Păsărică, R. G. Bozomitu, V. Cehan, and C. Rotariu, "Eye blinking detection to perform selection for an eye tracking system used in assistive technology," in *2016 IEEE 22nd International Symposium for Design and Technology in Electronic Packaging (SIITME)*, pp. 213–216, 2016.

[69] A. Królak and P. Strumillo, "Eye-blink detection system for human–computer interaction," *Universal Access in the Information Society*, vol. 11, pp. 1–11, 11 2011.

[70] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, pp. 770–778, 2016.

[71] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, 2017.

[72] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

[73] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[74] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.

[75] K. Cao, J. Ji, Z. Cao, C. Chang, and J. Niebles, "Few-shot video classification via temporal alignment," pp. 10615–10624, 2020.

[76] C. Doersch, A. Gupta, and A. Zisserman, "Crosstransformers: spatially-aware few-shot transfer," 2021.

[77] R. Hou, H. Chang, B. Ma, S. Shan, and X. Chen, "Cross attention network for few-shot classification," 2019.

[78] A. Raghu, M. Raghu, S. Bengio, and O. Vinyals, "Rapid learning or feature reuse? towards understanding the effectiveness of maml," 2020.

[79] T. Perrett, A. Masullo, T. Burghardt, M. Mirmehdi, and D. Damen, "Temporal-relational crosstransformers for few-shot action recognition," 2021.

[80] H. Zhang, L. Zhang, X. Qi, H. li, P. Torr, and P. Koniusz, "Few-shot action recognition with permutation-invariant attention," 2020.

[81] X. Zhu, A. Toisoul, J.-M. Perez-Rua, L. Zhang, B. Martinez, and T. Xiang, "Few-shot action recognition with prototype-centered attentive learning," 2021.

[82] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," 2017.

[83] K. Hara, H. Kataoka, and Y. Satoh, "Learning spatio-temporal features with 3d residual networks for action recognition," in *ICCV Workshops*, pp. 3154–3160, 10 2017.

[84] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," pp. 4489–4497, 01 2015.

[85] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," 2018.

[86] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," 2021.

[87] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?," pp. 6546–6555, 2018.

[88] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," 2012.

[89] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," 2011.

[90] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, F. Hoppe, C. Thurau, I. Bax, and R. Memisevic, "The "something something" video database for learning and evaluating visual common sense," pp. 5843–5851, 10 2017.

[91] T. Munkhdalai and H. Yu, "Meta networks," *Proceedings of machine learning research*, vol. 70, 03 2017.

[92] A. Santoro, S. Bartunov, M. M. Botvinick, D. Wierstra, and T. P. Lillicrap, "One-shot learning with memory-augmented neural networks," in *Proceedings of the International Conference on Machine Learning*, 2016.

[93] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," *CoRR*, 2017.

[94] L. Zhu and Y. Yang, "Compound memory networks for few-shot video classification," September 2018.

[95] Y. Yang and L. Zhu, "Label independent memory for semi-supervised few-shot video classification," vol. 44, pp. 273–285, 2022.

[96] M. Bishay, G. Zoumpourlis, and I. Patras, "Tarn: Temporal attentive relation network for few-shot and zero-shot action recognition," 2019.

[97] S. Dwivedi, V. Gupta, R. Mitra, S. Ahmed, and A. Jain, "Protogan: Towards few shot learning for action recognition," 2019.

[98] X. Li, Y. Zhang, C. Liu, B. Shuai, Y. Zhu, B. Brattoli, H. Chen, I. Marsic, and J. Tighe, "Vidtr: Video transformer without convolutions," 2021.

[99] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," vol. 30, 2017.

[100] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, p. 807–814, 2010.

[101] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, "A short note on the kinetics-700 human action dataset," *ArXiv*, 2019.

[102] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," 2018.

[103] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Val Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *ECCV*, 2016.

[104] H.-J. Ye, H. Hu, D.-C. Zhan, and F. Sha, "Few-shot learning via embedding adaptation with set-to-set functions," 2020.

[105] M. Monfort, A. Andonian, B. Zhou, K. Ramakrishnan, S. A. Bargal, T. Yan, L. Brown, Q. Fan, D. Gutfruend, C. Vondrick, *et al.*, "Moments in time dataset: one million videos for event understanding," 2019.

[106] Y. Yoshikawa, J. Lin, and A. Takeuchi, "Stair actions: A video dataset of everyday home actions," *ArXiv*, 2018.

[107] A. Kolesnikov, A. Dosovitskiy, D. Weissenborn, G. Heigold, J. Uszkoreit, L. Beyer, M. Minderer, M. Dehghani, N. Houlsby, S. Gelly, T. Unterthiner, and X. Zhai, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[108] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," pp. 843–852, 10 2017.

[109] R. Liu, A. A. Ramli, H. Zhang, E. Henricson, and X. Liu, *An Overview of Human Activity Recognition Using Wearable Sensors: Healthcare and Artificial Intelligence*, p. 1–14. Springer International Publishing, 2022.

[110] B. K. Akhmetzhanov, O. A. Gazizuly, Z. Nurlan, and N. Zhakiyev, "Integration of a video surveillance system into a smart home using the home assistant platform," in *2022 International Conference on Smart Information Systems and Technologies (SIST)*, pp. 1–5, 2022.

[111] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, p. 58443–58469, 2020.

[112] T. Iqbal, S. Rack, and L. D. Riek, "Movement coordination in human–robot teams: A dynamical systems approach," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 909–919, 2016.

[113] O. Durmaz Incel and S. Bursa, "On-device deep learning for mobile and wearable sensing applications: A review," *IEEE Sensors Journal*, vol. 23, no. 6, pp. 5501–5512, 2023.

[114] X. Liu, H. Zhang, H. Pirsiavash, and X. Liu, "Mastaf: A model-agnostic spatio-temporal attention fusion network for few-shot video classification," 2023.

[115] M. M. Islam and T. Iqbal, "Mumu: Cooperative multitask learning-based guided multimodal fusion," in *AAAI*, 02 2022.

[116] M. M. Islam and T. Iqbal, "Hamlet: A hierarchical multimodal attention-based human activity recognition algorithm," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10285–10292, 2020.

[117] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?," 2021.

[118] Z. Tong, Y. Song, J. Wang, and L. Wang, "VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training," in *Advances in Neural Information Processing Systems*, 2022.

[119] 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 *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[120] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal residual networks for video action recognition," in *NeurIPS*, 2016.

[121] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NeurIPS*, 2014.

[122] G. Liu, J. Qian, F. Wen, X. Zhu, R. Ying, and P. Liu, "Action recognition based on 3d skeleton and rgb frame fusion," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 258–264, 2019.

[123] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *CVPR*, 2019.

[124] X. Long, C. Gan, G. de Melo, X. Liu, Y. Li, F. Li, and S. Wen, "Multimodal keyless attention fusion for video classification," in *AAAI*, AAAI'18/IAAI'18/EAAI'18, AAAI Press, 2018.

[125] S. Zhang, Y. Yang, J. Xiao, X. Liu, Y. Yang, D. Xie, and Y. Zhuang, "Fusing geometric features for skeleton-based action recognition using multilayer lstm networks," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2330–2343, 2018.

[126] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," pp. 1096–1103, 01 2008.

[127] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[128] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," in *ICLR*, 2022.

[129] R. Wang, D. Chen, Z. Wu, Y. Chen, X. Dai, M. Liu, Y.-G. Jiang, L. Zhou, and L. Yuan, "Bevt: Bert pretraining of video transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12 2021.

[130] H. Tan, J. Lei, T. Wolf, and M. Bansal, "Vimpac: Video pre-training via masked token prediction and contrastive learning," *ArXiv*, vol. abs/2106.11250, 2021.

[131] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *NeurIPS*, 2020.

[132] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019.

[133] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[134] L. Wang, B. Huang, Z. Zhao, Z. Tong, Y. He, Y. Wang, Y. Wang, and Y. Qiao, "Videomae v2: Scaling video masked autoencoders with dual masking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14549–14560, June 2023.

[135] S. Zhang, J. Zhou, and X. He, "Learning implicit temporal alignment for few-shot video classification," 2021.

[136] R. Y. Aminabadi, S. Rajbhandari, A. A. Awan, C. Li, D. Li, E. Zheng, O. Ruwase, S. Smith, M. Zhang, J. Rasley, and Y. He, "Deepspeed- inference: Enabling efficient inference of transformer models at unprecedented scale," in *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15, 2022.

[137] Y. Guan and T. Ploetz, "Ensembles of deep lstm learners for activity recognition using wearables," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, 03 2017.

[138] N. Hammerla, S. Halloran, and T. Ploetz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," 04 2016.

[139] V. Murahari and T. Ploetz, "On attention models for human activity recognition," pp. 100–103, 10 2018.

[140] H. Zhang, Z. Xiao, J. Wang, F. Li, and E. Szczerbicki, "A novel iot-perceptive human activity recognition (har) approach using multihead convolutional attention," *IEEE Internet of Things Journal*, vol. PP, pp. 1–1, 10 2019.

[141] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, vol. 8, pp. 193907–193934, 2020.

[142] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural Computation*, vol. 16, no. 12, pp. 2639–2664, 2004.

[143] J. Wang, T. Zhu, J. Gan, L. Chen, H. Ning, and Y. Wan, "Sensor data augmentation by resampling for contrastive learning in human activity recognition," 2021.