Skeleton-based fall detection using Computer Vision

Duong Thi Phuong Dung
Pham Le Anh Duc
Can Thi Hoa Mai

Supervisor: Assoc. Prof. Phan Duy Hung

Bachelor of Artificial Intelligence
Hoa Lac campus - FPT University
2023

Acknowledgment

We would like to thank Assoc. Prof. Phan Duy Hung, our especially enthusiastic supervisor, for helping us to develop our ideas and providing us with precious guidance throughout the last 4 months.

We would like to thank all lecturers at FPT University and particularly those at the Information Technology Specialization Department for helping us gain the understanding and background necessary to conduct this thesis.

We also would like to appreciate our families and friends for always encouraging us to do our best and giving us useful comments to complete this thesis.

Abstract

Falling is one of the biggest public health issues that can cause many serious long-term repercussions for patients and their families. In this thesis, we propose an appropriate model for fall detection using graph convolutional networks. Recently, most problems related to human action recognition, including fall detection, can be handled by applying the Spatial Temporal Graph Convolutional Networks model (ST-GCN) using 2D or 3D skeletal data. We take advantage of the transfer learning technique from the NTU RGB+D consisting of 60 daily actions to extract features for the fall detection task efficiently. Besides, to highlight critical frames in the original sequence, we suggest using a temporal attention module. This module consists of two parts: (1) average global pooling, and (2) two fully connected layers to generate an attention score for each frame. We perform experiments on two datasets, i.e., FallFree and TST v2. This leads to a 3.12% increase in the TST dataset and a 2.67% improvement in the FallFree dataset. Notably, with respect to FallFree, the accuracy of the model is up to 100%.

Keywords: Fall Detection, Human Action Recognition, Spatial Temporal Graph Convolutional Networks, Attention Mechanism

Table of contents

List of tables

List of figures

List of abbreviations and acronyms

GCN:	Graph Convolutional Networks
ST-GCN:	Spatial Temporal Graph Convolutional Networks
RGB:	Red Green Blue color model
CNN:	Convolutional Neural Networks
RNN:	Recurrent Neural Networks
AGCN:	Adaptive Graph Convolutional Networks
MS-AWGCN:	Multi-scale Skeleton Adaptive Weighted Graph Convolution Networks
RAM:	Recurrent attention mechanism
SEnet:	Squeeze and Excitation Network
SE:	Squeeze and Excitation
CBAM:	Convolution Block Attention Module
MP:	Max Pooling
GAP:	Global Average Pooling
LSTM:	Long Short Term Memory
FCN:	Fully Convolutional Network
IMU:	Inertial Measurement Unit
ADL:	Activities of Daily Living
XEF:	Extend Event File
GPU RAM:	Graphics Processing Unit Random Access Memory
SGD:	Stochastic Gradient Descent
ROC AUC:	Receiver Operating Characteristic Area Under the Curve
FP:	False Positive
SVM:	Support Vector Machine
CUSUM:	Cumulative Sum
NCA:	Neighborhood Component Analysis
RF:	Random Forest

1. Introduction

1.1. Motivation

Falling is one of the major public health problems. Each year, an estimated 684,000 people die from falls, making it the second largest cause of unintentional injury death, after road traffic injuries. Besides, it also leads to more years of disability in life than traffic accidents, drowning, burns, and poisoning combined [1]. Falling accidents not only cause direct consequences for the patient but also greatly affect the family members. To reduce and prevent falls and their related consequences, a variety of methods have been developed to monitor, detect as soon as possible, and alert related personnel of patients timely. Especially for families who often have elderly people or children at home alone, fall detection becomes an important task in both healthcare and daily life.

1.2. Related work

1.2.1. Fall detection & Human action recognition

It can be seen that much research and application about fall detection has been performed and has obtained good results. The three main types of fall detection methods are video-based, ambience device-based, and wearable device-based. In the wearable device-based approach, almost any part of the human body can have the sensors mounted, while in the ambient device-based method, the sensors are installed in the living space of the person being followed, such as walls, floors, beds, etc. Take the Apple watch as an example, from series 4, Apple gathered data from 2,500 people wearing this product and had more than 250,000 days of data for fall detection [2]. The main sensors applied to detect falls are the accelerometer and the gyroscope. The accelerometer can measure a higher amount of gravity forces (16Gs - 32Gs) and the gyroscope turned on 24/7 can measure the rate of rotation, and visualize the different ways it does this through three axes in space. Figure 1 is an example of fall detection in Apple Watch, if a fall is detected, it first confirms the patient's condition before making an emergency call. However, not everyone has the condition and feels comfortable when wearing devices all the time. Hence, the video-based approach, including RGB-based (raw video) and skeleton-based, from indoor cameras for fall detection applications is being widely developed. When compared to the RGB-based approach, the skeleton-based method has the huge advantage of not only computational cost but also not being affected by changes in the surrounding environment. Extracting skeleton points from video helps to recognize human action

in general and to detect falls in particular. Therefore, it is possible to take advantage of the research results on human action recognition in the fall detection problem.

Figure 1. An example of fall detection on the Apple Watch [2].

Recently, there have been leaps and bounds in human action recognition research from CNN, and RNN to graph convolutional networks (GCN). In this section, papers related to GCN will be focused on reviewing. GCN includes two types: spectral and spatial. While spectral GCN transforms graphs into the spectrum domain and employs the graph Fourier transform, spatial GCN gathers information from nearby nodes [3]. In 2018, Yan et al. presented a "Spatio-temporal graph convolution network" (ST-GCN) that is much more advanced than old methods by learning both the space and time in the data. Figure 2 displays the pipeline of the ST-GCN model presented in [4] which consists of the graph representation of the human body and ST-GCN blocks to extract spatial and temporal features. Each human skeleton's spatial characteristics are extracted using a GCN, and the same joint's continuous time edge is subjected to a time convolution network [4]. Original ST-GCN is created by ST-GCN blocks that alternately apply temporal and spatial graph convolutions to a skeleton graph [3]. Finally, the action class is predicted by fully connected layers and a classifier using softmax.

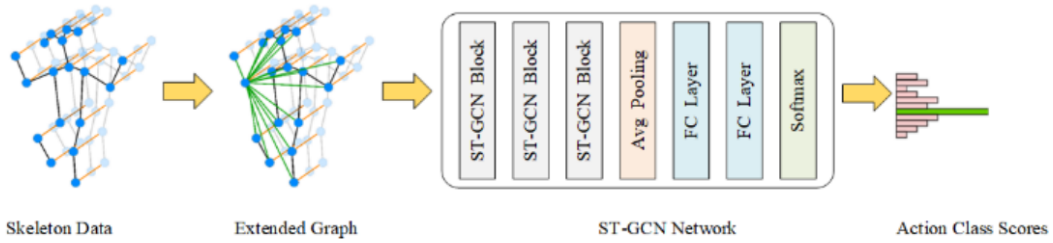


Figure 2. Example of a system using ST-GCN to recognize human action through skeleton-based [5].

In the last few years, many researchers have made great improvements based on ST-GCN. An impressive research was adaptive graph convolution neural networks (AGCN) by Wang et al. [6]. Their GCN has some improvements such as constructing a unique graph Laplacian, learning distance metric for graph update, feature embedding in convolution, accept flexible graph inputs. Xu et al. proposed the "Multi-scale skeleton adaptive weighted graph convolution networks" (MS-AWGCN) that establish a model complete understanding of the relationships between human motion with a fresh depiction of the human body: divided into ten parts, each of them is extracted spatial and temporal feature by using ST-GCN, finally all features are integrated with the feature of the body skeleton. Then, for each sampling strategy, they apply a learning-weighted strategy that enhances the features while combining [7]. However, it still has the challenge of keeping stable or raising the accuracy and simultaneously simplifying the GCN model.

With respect to the fall detection problem, Oussema Keskes et al. introduced a general "Vision-based fall detection using ST-GCN" [8]. They encoded temporal information and used ST-GCN to manage the spatial arrangement and temporal dynamics of joints. Additionally, they employed the transfer learning technique to aid the model in utilizing features extracted from the task of human action recognition, which is related to fall detection. The system showed effectiveness when having good performance on 2 fall datasets with the accuracy of some experiments up to 100%. Figure 3 depicts an application of real-time fall detection using deep learning and advanced image processing techniques with an accuracy of 85% [9].

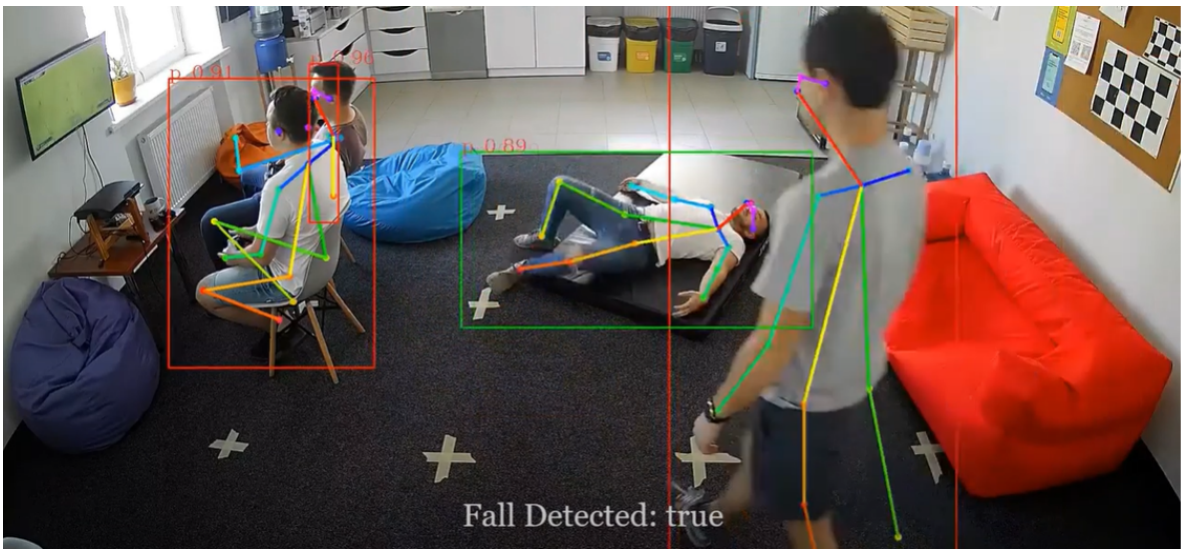


Figure 3. An 85% accurate video analytics solution for managing slip and fall accidents was produced by Abto Software [9].

1.2.2. Attention mechanism

During the research process, we found out that the attention mechanism may be useful for our problem of fall detection. The method that shifts the focus to the most critical area of the scene while simultaneously blurring the surrounding parts is called the attention mechanism. The human visual processing system serves as an inspiration for this method. Figure 4 briefly describes how a human's brain and eyes work in reality, for example, when you read a book. Your brain directs attention to the small region you are currently reading and ignores the other parts.

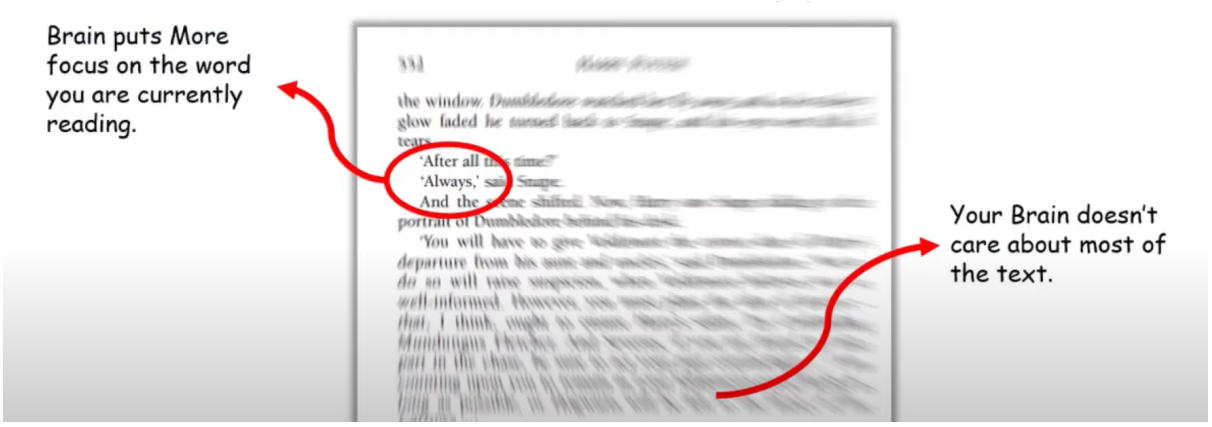
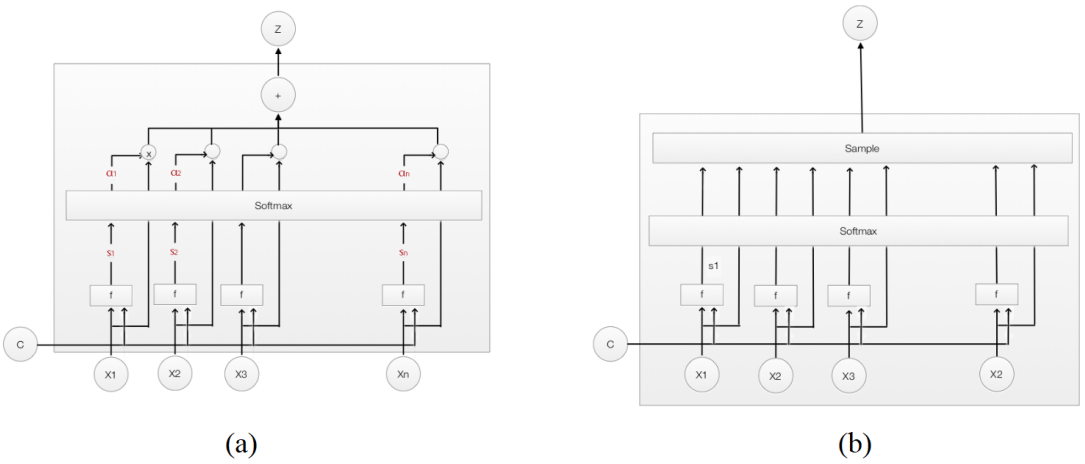


Figure 4. An example of the human visual processing system [10].

Researchers have taken advantage of this mechanism in many problems that have sequence data such as machine translation, image captioning, human action recognition, etc. The attention mechanism helps the model to assign different weights for each input to represent different levels of importance of the input sequences as illustrated in Figure 5. Especially, for fall detection problems where some critical frames have abrupt changes in action [5], it is expected that the attention mechanism aids the model in reasonably weighing the sequence of skeletal frames.

Figure 5. An example of assigning different weights to each input in a data sequence [10].

Recently, attention modules have received extensive research and have been widely used in many applications. Attention mechanisms for computer vision can be classified as soft attention and hard attention [11]. Figure 6 shows the differences between soft attention and hard attention. Which, soft attention computes a weighted input from input features so that it is a differentiable function, and the weights of the network can learn from forward and backward propagation [12]. Hard attention, on the other hand, Xu et al. [11] use attention scores to select a single feature such as via argmax function, which makes it not differentiable, and therefore, it is often supported by using reinforcement learning. Employing hard attention, Mnih et al. [13] presented the recurrent attention mechanism (RAM) that combines the RNN and reinforcement learning techniques. RAM takes a glimpse window as its input, then via the network, it selects the next location to focus on (important regions).



(a) (b)

Figure 6. Examples of attention: (a) soft attention that calculates context vector (z) as the weighted sum of all input vectors (x_i), (b) hard attention that uses sample function to select context vector from all input vectors [14].

With respect to soft attention, there are three well-known types, i.e., channel attention, spatial attention, and temporal attention [12, 15]. In image processing, each channel contains specific information about distinct objects. Via CNN, the number of channels may change in every layer and generate new information. A novel model called SEnet (squeeze and excitation networks) is presented by Hu et al. [16] to highlight channels we need to pay attention. First, the input feature is fed into a squeeze block to extract global features by using global average pooling (GAP). Then, through fully connected layers (FC) in the excitation block, the model then generates a weight for each channel.

However, channel attention usually ignores some important features of spatial attention. For that reason, mixed attention is now studied and applied in many computer vision applications. For example, in [17], Woo et al. introduce the hybrid of spatial and channel attention. They exploit the SE block for channel attention by adding a max pooling (MP) beside the GAP in the original work. Then, for spatial attention, they also use GAP and MP along the channel axis to highlight informative regions [17]. In another case, a mix of spatial and temporal attention is proposed in an "End-to-end spatio-temporal attention model for human action recognition from skeleton data" [18]. They develop three main modules, including a spatial attention block to learn essential features of human joints, an RNN block with the LSTM layer, and finally, a temporal attention block to stay focused on key frames for action recognition. Recently, Zhu et al. [5] used the collaboration of channel attention and frame attention (temporal attention). They are supposed to learn the correlations among channels in the time dimension and extract information about key frames by adding a squeeze-and-excitation (SE) block with the temporal gated unit and attention

block, respectively. We can apply that idea to our problem of fall detection, which needs to pay attention to frames having significant changes to recognize the action [5].

In summary, both fields of research show good performance and potential for the problem of fall detection. However, there is not much existing research that combines these two techniques on skeletal data, especially the temporal attention mechanism and sees how it works. Thus, we are trying to integrate human action recognition using ST-GCN with a transfer learning technique and attention mechanism to extract crucial features from skeleton data.

1.3. Contribution

The main contribution of this thesis is to build a model to find an appropriate method for the fall detection problem. The combination of ST-GCN with the temporal attention mechanism was applied, in which ST-GCN automatically learns not only spatial but also temporal patterns from skeletal data while the attention mechanism helps to improve the extraction of spatio-temporal features of a skeleton sequence by considering different importance levels of frames.

As an additional contribution, the transfer learning technique is applied to leverage the available knowledge in extracting features from human action recognition tasks to help with fall detection problems. Therefore, in our proposed method, the learned knowledge is employed so that there is no need to retrain from scratch with new data and still provide high accuracy.

We tested our recommended model to verify its performance. The study experimented on two fall datasets, TST v2 and FallFree, and then compared with several methods for the best results with up to 100% accuracy.

2. Methodology

2.1. Overview pipeline

In this thesis, we propose the following pipeline for our problem of fall detection. Figure 7 illustrates the pipeline's two phases: train and fine-tune on the fall datasets based on the pre-trained ST-GCN model; apply temporal attention and determine the output by the classifier layer.

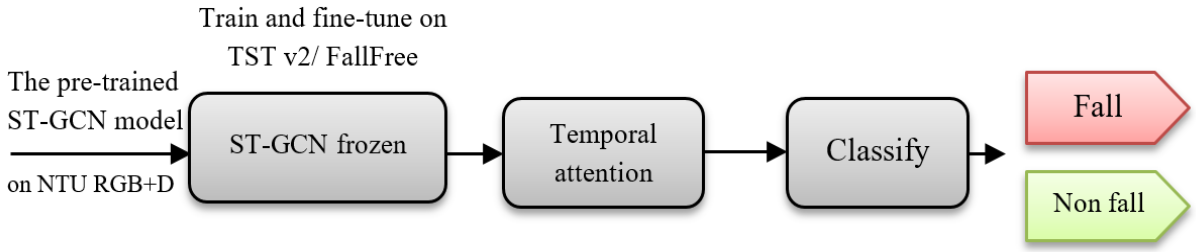


Figure 7. The structure of our proposed fall detection model.

Our proposal method concentrates on experimenting with the efficiency of the temporal module when combined with the ST-GCN model for fall detection. In the next sections, we will explain the details of each step more clearly.

2.2. Spatial temporal graph convolutional networks

The key idea behind ST-GCN is to model the spatiotemporal relationships between various components of the input data by means of graph convolutional neural networks (GCNs). Each human joint in a skeleton sequence is typically represented by 2D or 3D coordinates in each frame. In this study, we create hierarchical representations of the skeletal sequences using the spatial-temporal graph with the input of $N \times C \times T \times V \times M$, where N is the batch size, C denotes the number of channels, T is the length of sequences (frames), V is the number of graph nodes, and M indicates the number of instances. In [4], Yan et al. used a skeleton sequence with N joints and T frames that has both intra-body connectivities (E_S) referring to the relationships between different joints in the same frame and inter-frame connectivity (E_F) referring to the relationships between the joint positions in different frames. They constructed an undirected spatial-temporal graph $G = (V, E)$, in particular, V denoting the set including all joints in a skeleton sequence: $V = \{v_{ti} \mid t = 1, ..., T;\ i = 1, ..., N\}$ and edge set E consists of two parts: 1) $E_S = \{v_{ti}, v_{tj} \mid (i, j) \in H\}$, where H is the group of the human body's naturally linked joint pairs; and 2) $E_F = \{v_{ti} v_{(t+1)i}\}$.

First, the GCN is sort of different from the normal CNN where we just use weights of a kernel to map with the fixed neighbors of the current pixel. In GCN, for each node, it is necessary to define its neighbor nodes and label the weight coefficient for each of them. As can be seen in Figure 8, the authors proposed 3 methods for the graph labeling process, i.e., uni-labeling partition, distance partition, and spatial configuration partition. In this work, they used just a 1-neighbor set of joint nodes for

each current node to convolve. For each neighbor node around the root v_{tj}, they define function l_{ti} to map it with a unique label.

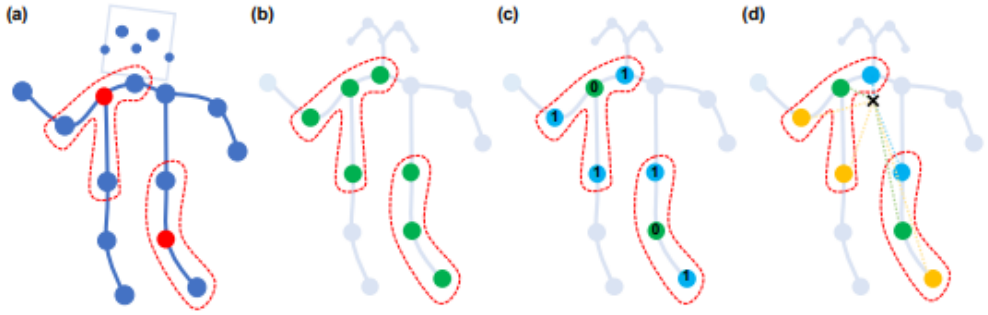


Figure 8. (a) an illustration of an input skeleton frame; (b) Uni-labeling partition; (c) Distance partitioning; (d) Spatial configuration partitioning [4].

In the uni-labeling partition, all nodes have the same label $l_{ti}(v_{tj}) = 0$ $\forall i, j \in V$. In distance partition, $l_{ti}(v_{tj}) = 0$ for the root node, and 1 for the neighbor nodes. Finally is spatial configuration partition, because the body skeleton is restricted in spatial, they may use this special spatial arrangement in the partitioning process. Yan et al. come up with a strategy to split the neighbor set into three groups: The root node itself, the centripetal group, which includes the adjacent node positioned closer to the skeleton's gravity center, and the centrifugal group. Note that the gravity center is defined as the average coordinate of all joints at a frame. Properly, the nodes are labeled by the distance to the skeleton gravity and we have:

$$l_{ti}(v_tj) = \begin{cases} 0 & \text{if } r_j = r_i \\ 1 & \text{if } r_j < r_i \\ 2 & \text{if } r_j > r_i \end{cases}$$

Based on these partitioning strategies, the GCN is implemented to extract from both temporal and spatial features. Then, extracted features are provided into the classifier layer to predict the class of action.

2.3. The pre-trained ST-GCN model

Recently, the problem of human action recognition has become more popular and the ST-GCN model applied in almost articles can prove its outstanding performance. Therefore, many studies were conducted and datasets were introduced to

solve that problem. In [4], Yan et al. provide the pre-trained ST-GCN model on some datasets, so Keskes et al. [8] suggested a method that can exploit the huge data for human action recognition, including fall samples, into the problem of fall detection and demonstrated that it is effective on FallFree and TST v2 dataset. Figure 9 describes the components of the ST-GCN model which is a list of 10 layers of ST-GCN units and a fully connected layer for a classifier. According to [8], they froze the first 9 layers of the second component (ST-GCN networks) and changed the number of classes in the output layers to 2 (for fall and no fall). The data was split based on the cross-subject evaluation method. Inspired by that idea, we decide to apply the pre-trained ST-GCN model on the NTU RGB+D dataset [4], then carried out new training and fine-tuning with TST v2 and FallFree datasets as the first step in our pipeline for detecting falls.

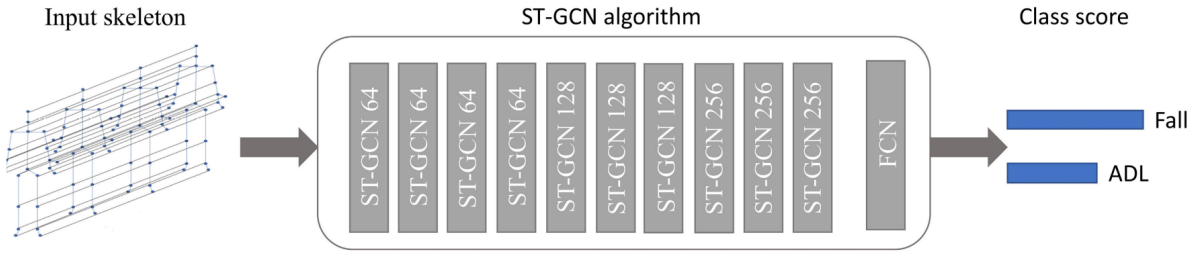


Figure 9. ST-GCN model architecture [8].

2.4. Attention mechanism

As noted above, not all frames have the same level of importance when recognizing human actions. In particular, in fall detection, some frames have sudden changes in movement, and it is important to pay more attention to those frames. Therefore, inspired by a "weakly-supervised temporal attention module" in [19], we propose to involve a temporal attention mechanism in our problem to highlight critical frames and to detect falls more accurately. As depicted in Figure 10, our temporal attention module includes an average pooling layer and two fully connected layers ending up with the Softmax activation function. The average pooling layer takes an input of the size $N \times C \times T \times V$ and compresses and extracts global information from each frame into the size of $N \times T \times 1 \times 1$. This layer also helps to downsize the number of parameters when fitting in the fully connected layers [5].

Two fully connected layers generate T features to learn the appropriate weights of each input frame [19]. Then, the output of the second FC layer is applied softmax to compute a distribution of weights with the same shape. In the original model [19], the

authors use Sigmoid to calculate the weight matrix and exploit it as the relationship between frames. However, if very large or small values appear frequently in the data, sigmoid then returns all weights of approximately 1 or 0. This will bring us senseless value for emphasizing essential frames, so we decide to employ softmax instead. The distribution of weights (or attention scores) of each frame is mapped into the range (0, 1) to show different levels of importance in the whole frame sequence. We multiply this weight distribution with the initial input to adjust attention scores to each frame. Finally, it is scaled by factor α and added back to the input information.

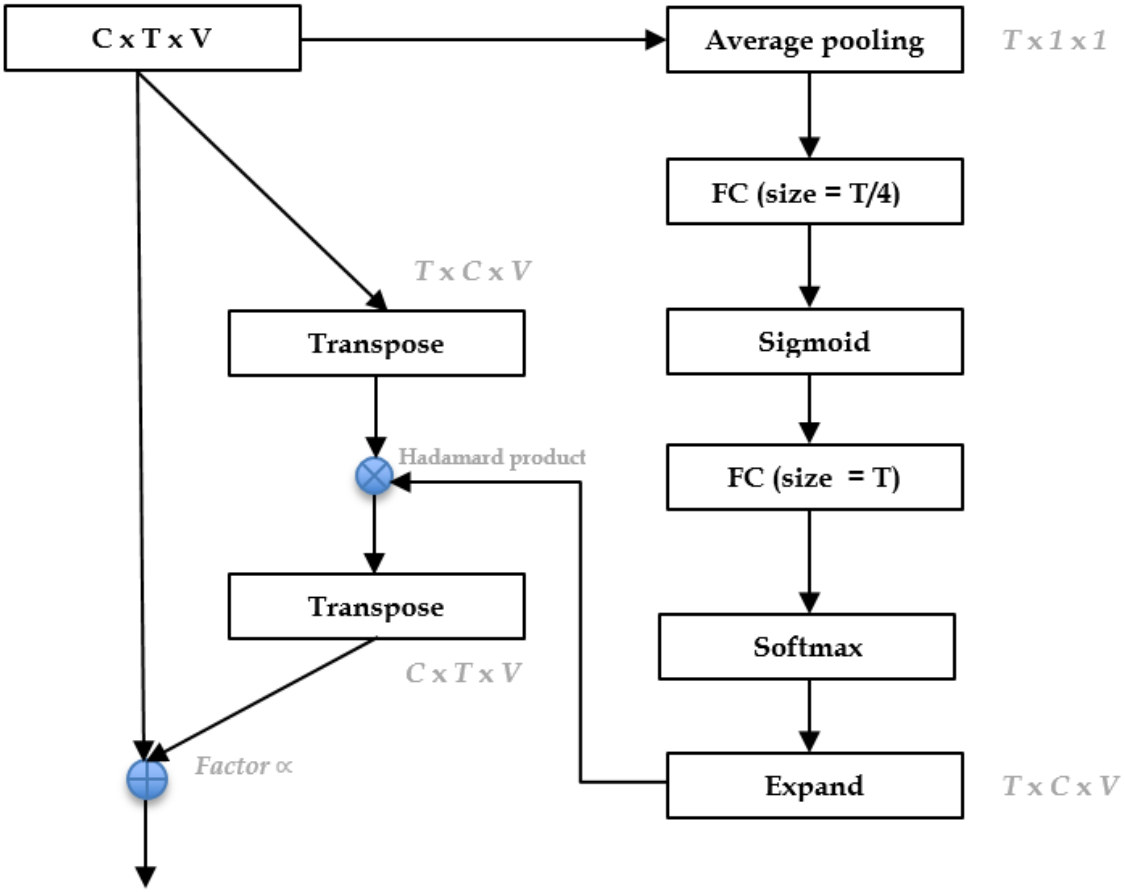


Figure 10. Structure of the temporal attention module.

3. Experiment

In this section, two fall detection datasets TST v2 [20] and FallFree [21] are used to train and evaluate the model. The pre-trained ST-GCN model on the NTU RGB+D dataset [4] was utilized to extract features for the specific task of fall detection more efficiently.

3.1. Datasets & Evaluation metrics

3.1.1. Datasets

Regarding human action recognition, the most common dataset may include the NTU RGB+D [22]. It covers 60 action classes (40 everyday acts, 11 mutual actions, and 9 health-related actions) and 56,880 video samples of 40 distinct subjects. The dataset comprises infrared videos, RGB videos, 3D skeletal data of 25 main body joints that can be seen in Figure 11, and depth map sequences for each sample. All samples are captured by using three Kinetics cameras concurrently at different horizontal angles, i.e., - 45°, 0°, 45° respectively. For evaluation, they define 2 types of action classification criteria. The first is the cross-subject evaluation in which the training and test set is split by the subject's number. The second one is the cross-view evaluation, in which the training set includes samples of cameras 2 and 3 while the test set uses data that is recorded in camera 1.

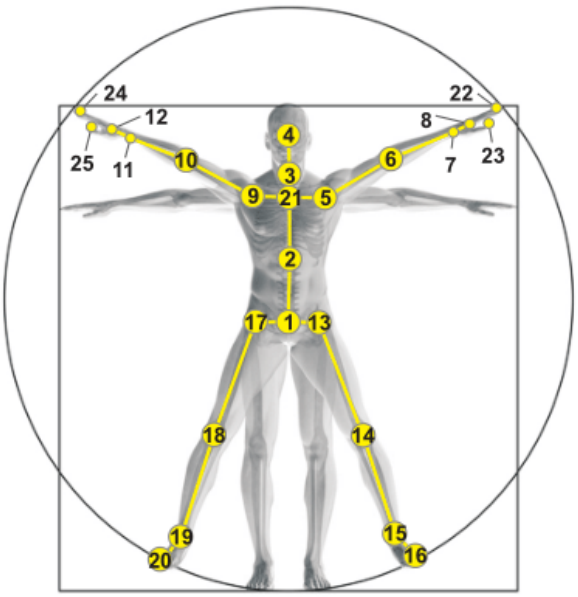


Figure 11. Illustration of 25 body joints in the 3D skeleton data [22].

The TST fall detection dataset v2 [20] was published in 2015 for the specific task of fall detection. The data was gathered by using the Microsoft Kinect v2 camera with the inertial measurement unit (IMU), the device mounted on the actor's waist and wrist. It provides three kinds of data: depth, skeletal, and acceleration data. This dataset is split into two groups: activities of daily living (ADL) and the fall. There are

four actions in each category. Sitting, grasping, walking, and laying belong to the ADL category, whereas the front fall, the backward fall, the lateral fall, and the ends-up sitting fall in the fall category. Each action was simulated by 11 actors between the ages of 22 and 39 with heights ranging from 1.62 to 1.97 meters. They performed all actions three times. Consequently, the total number of samples in this dataset is 264, each category is composed of 132 samples.

Alzahrani et al. [21] introduced the FallFree dataset in 2017. They also use a Microsoft Kinect v2 camera to collect data. The data videos are provided as extended event files (XEF). There are five types of data stored in FallFree that are color, skeleton, depth, infrared, and body index. The dataset contains 79 actions equivalent to 391 samples. True fall, pseudo fall, and ADL are the three major categories corresponding with 208, 115, and 68 samples. First, the true fall category is made up of forward, backward, and sideways falls. Second, falls with recovery or falls due to syncope are included in the pseudo-fall category. A fall with recovery is the act of a person losing balance, then obtaining balance, whereas a fall due to syncope is when a person becomes unbalanced and tries to cling to the wall. And the last category, ADL, is composed of daily life movements. Two actors performed the actions in three separate rooms with dissimilar lighting prerequisites. They were 30 and 35 years old with a height of 1.50 and 1.68 meters respectively. The first actor repeated each action four times while the other performed most of the actions only once.

Table 1. Descriptions of the datasets

Dataset	# Actions	# Subjects	# Fall samples	# Samples	Publish year
NTU RGB+D	60	40	276	56,880	2016
TST v2	5	11	132	264	2015
FallFree	10	2	208	391	2017

This thesis only uses 3D skeletal data of the dataset for training and testing due to the advantages of storage capacity and training time. These skeleton sequences are all provided by the authors, so extracting the skeletons again from the raw video is not necessary. Regardless, the FallFree dataset needs an additional tool to read the XEF file format and transform it into a skeleton file. In [23], Issac introduces a KineticXEFTools library built-in .NET framework. It supplies two Visual Studio solutions, i.e., KineticXEFTools solution for reading XEF files and XEFExtract solution to extract desirable formats including body skeleton files.

3.1.2. Evaluation metrics

We use the following typical metrics used for evaluating binary classifiers to the fall detection problem:

Accuracy: describes the percentage of rightly predicted samples out of the total number of samples.

$$\text{Accuracy} = \frac{TP + TN}{\text{total sample}} = \frac{TP + TN}{TP + TN + FP + FN}$$

Sensitivity: measures the rate of correctly predicting positive samples across all samples belonging to the positive group.

$$\text{Sensitivity} = \frac{TP}{\text{total actual positive}} = \frac{TP}{TP + FN}$$

Specificity: measures the ratio of rightly predicting negative samples to all actually negative samples.

$$\text{Specificity} = \frac{TN}{\text{total actual negative}} = \frac{TN}{TN + FP}$$

False positive rate (FPR): describes the rate of incorrectly predicted samples from actual negative to positive over the total number of actual negative samples.

$$\text{FPR} = \frac{FP}{\text{total actual negative}} = \frac{FP}{TN + FP}$$

F1-score: is the harmonic mean between precision and recall.

$$\text{F1} = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}} = \frac{TP}{TP + \frac{FP + FN}{2}}$$

ROC AUC: describes the area under the ROC curve.

where:

TP: stands for true positive, meaning the prediction is "fall", the label is also "fall".
FP: stands for false positive, meaning the prediction is "fall" but the label is "no fall".
TN: stands for true negative, meaning the prediction is "no fall", the label is "no fall".
FN: stands for false negative, meaning the prediction is "no fall" but the label is "fall".

3.2. Experiment settings

Our research is implemented in Python and PyTorch frameworks. We perform experiments on different machines with 2 types of NVIDIA cards: Geforce GTX 1650

and Geforce MX130. The training process is applied by following steps. First, we pre-process data and set the maximum number of frames to 300, i.e., zero-padding when the original video has less than 300 frames while with videos having more than 300, we remove redundant frames (from the end of videos). For better performance, we apply the pre-trained model provided in [4] to two datasets for fall detection. Then, we perform the ST-GCN module to extract the temporal and spatial features simultaneously. After that, extracted features are put into the temporal attention module to highlight critical frames in order to detect falls. The kernel size for the ST-GCN is set to 9, and other configurations are the same as in [8]. The factor α in the attention module, via experiments, is set to 0.5 for the highest result. Due to the limit of GPU RAM, we train in 50 epochs and the batch size is 8. Finally, to enhance the model's performance, we adjust the SGD optimizer's base learning rate to 0.1.

3.3. Results & Analysis

Our model was carried out through two experiments that correspond with two datasets. The TST v2 dataset was split into two subsets. The training set contains videos that were performed by 7 actors: 1, 3, 5, 7, 9, 10, and 11. The videos of the 4 remaining actors were a test set. The FallFree data videos were performed by two people. While the first person simulated all actions four times, the other did once. Due to its characteristics, the videos of the first and the second subject were used for training and testing, respectively.

The first experiment gave acceptable results with the TST v2 dataset. The values of accuracy, sensitivity, specificity, FPR, F1-score, and ROC AUC were acquired at 89.58%, 97.22%, 85%, 12.5%, 87.5%, and 91.11%, respectively. The second experiment had good results when using the FallFree dataset. The achieved results were 100% accuracy, 100% sensitivity, 100% specificity, 0% FPR, 100% F1-score, and 100% ROC AUC.

Table 2. Results of our proposed method on two fall datasets

Dataset	Accuracy	Sensitivity	Specificity	FPR	F1-score	ROC AUC
TST v2	89.58%	97.22%	85%	12.5%	87.5%	91.11%
FallFree	100%	100%	100%	0%	100%	100%

Our proposed method was compared with several methods in Tables 3 and 4, presenting the reached results on the TST v2 and FallFree dataset, respectively. Both used datasets store some kind of data such as RGB videos, depth frames, acceleration streams, skeleton joints, infrared images, etc. Nevertheless, we only used skeleton data for our method, hence we decided to compare with methods that also used skeleton instead of the other data types of the two fall datasets.

Table 3. Comparison of our method and others on the TST v2 dataset

Method	Algorithms	Evaluation method	Accuracy
[24]	SVM	2/3 of data for training, 1/3 of data for testing	92.05%
[25]	SVM	70% of data for training, 30% of data for testing	93.56%
[26]	SVM and CUSUM	Leave-one-person-out	91.7%
[8]	Pre-trained ST-GCN	Cross subject	100%
[8]*	Pre-trained ST-GCN	Cross subject	86.46%
Our method	**Pre-trained ST-GCN and attention mechanism**	**Cross subject**	**89.58%**

* The result we obtained after implementing the method in [8]

On the TST v2 dataset, we compare our method with four others. As can be seen in Table 3, our approach is inferior to others. We applied the idea of using a pre-trained ST-GCN model in [8] to our method. However, in the process of implementing this idea, we did not achieve the results they published on the TST v2 dataset (this did not happen with the FallFree dataset). Compared to their 100% accuracy, we only received an accuracy of 86.46%. Their study does not mention how the datasets were pre-processed. Thus, discrepancies between our results and theirs may occur due to the way we pre-processed data was not the same as they did. After combining with the attention mechanism we proposed, the accuracy of our model increased by 3.12%.

The FallFree dataset has been public since 2017, thus the number of studies that utilized it has not been large. Table 4 compares our method with two models on the FallFree dataset. Compared to the model in [27], our accuracy is 0.5% higher. Our approach adopted the idea of [8] that uses the pre-trained ST-GCN model, in addition to the attention mechanism. It resulted in our model giving 2.67% better accuracy.

Table 4. Comparison of our method and others on the FallFree dataset

Method	Algorithms	Evaluation method	Accuracy
[27]	NCA for feature selection and RF classifier	70% of data for training, 30% of data for testing	99.5%
[8]	Pre-trained ST-GCN	Cross subject	97.33%
Our method	**Pre-trained ST-GCN and attention mechanism**	**Cross subject**	**100%**

In Figure 12, which goes from frame 20 to frame 45, the areas that have brighter colors mean "when to pay attention", since, in those frames, changes in movement concur very quickly. The falling action is fast and can last just 1 second, so the number of frames to focus on is about 30. Frames before the fall starts and after it ends have little differences, thus acquiring a lower score and dark color in the distribution map.

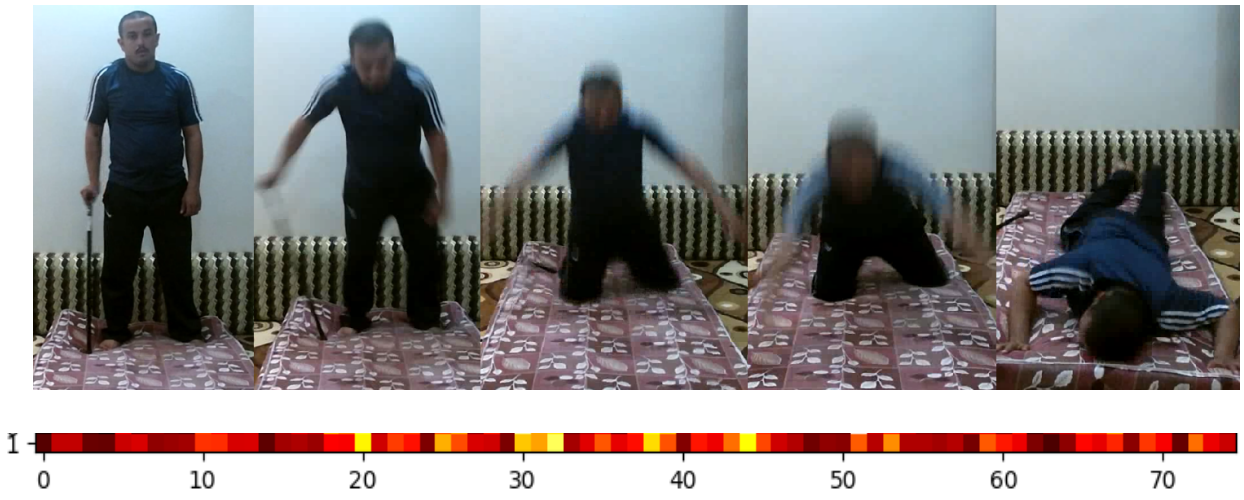
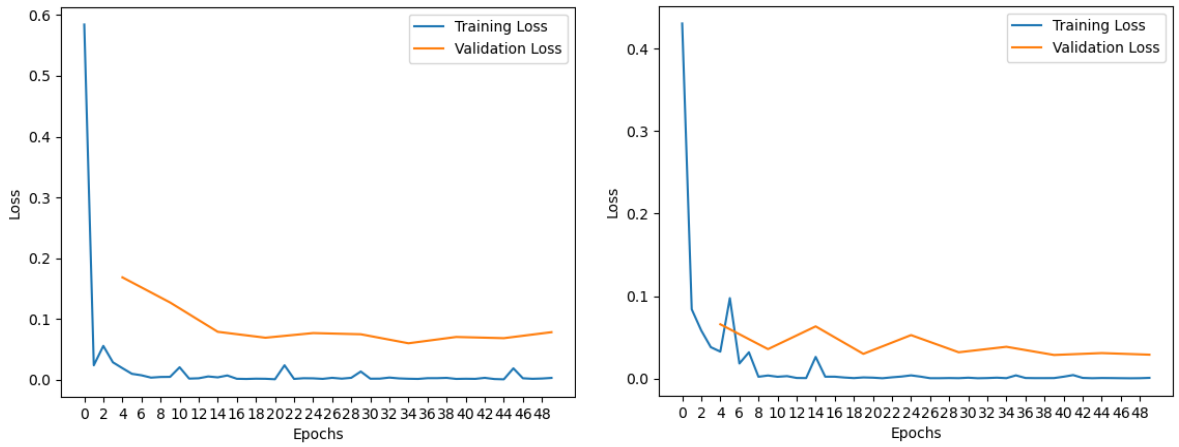


Figure 12. The falling process and the distribution of attention weights for a 75-frame sequence.



(a) (b)

Figure 13. The training and validation loss graph in cases: (a) without and (b) with our proposed attention mechanism on the FallFree dataset.

As shown in Figure 13, the "distance" between the blue line and orange line in the case with an attention mechanism is lower than in the case without an attention mechanism. In Figure 13(b), the training loss in the behind epochs is more "stable" than the remaining case. In Figure 13(a), the validation loss values are approximately around 0.1 in later epochs, whereas in Figure 13(b), the entire orange line is completely under the 0.1 value. It can be seen that the attention technique reduces loss and makes the learning process more effective.

4. Conclusion & Future Work

This thesis proposes a model for fall detection by combining the pre-trained ST-GCN model and the temporal attention technique. Using the pre-trained model helps us gain knowledge of extracting features in both spatial and temporal dimensions from human action recognition task. Besides, the temporal attention mechanism enhances the model's performance by highlighting important frames in a video. Experiment results demonstrate the effectiveness of our proposed method when performing on the TST v2 and FallFree datasets. Especially in the experiment with the FallFree dataset, the result is extremely good when obtaining an accuracy of 100%. It also leads to over a 3% improvement in the TST v2 dataset.

However, our proposed method's result on the TST v2 dataset is currently fairly poor compared to other methods. In the future, we are trying to figure out the causes and make improvements to this dataset. Moreover, we also want to do more experiments on different kinds of attention such as spatial attention, channel attention,

etc., and put them all together to see their effectiveness for our problem. Finally, it is strongly essential to detect falls as soon as possible, hence, our next goal is to develop the current model into a real-time fall detection application that can detect falls timely and send an urgent warning to their relatives.

References

1. Falls. (n.d.). Who.int. Retrieved February 4, 2023, from https://www.who.int/news-room/fact-sheets/detail/falls
2. Verger, R. (2018, October 3). The Apple Watch learned to detect falls using data from real human mishaps. Popular Science. https://www.popsci.com/apple-watch-fall-detection/
3. Wang, Q., Zhang, K., & Asghar, M. A. (2022). Skeleton-based ST-GCN for human action recognition with extended skeleton graph and partitioning strategy. IEEE Access: Practical Innovations, Open Solutions, 10, 41403–41410. https://doi.org/10.1109/access.2022.3164711
4. Yan, S., Xiong, Y., & Lin, D. (2018). Spatial Temporal Graph Convolutional Networks for skeleton-based action recognition. https://doi.org/10.48550/ARXIV.1801.07455
5. Zhu, Q., Deng, H., & Wang, K. (2022). Skeleton action recognition based on temporal gated unit and adaptive graph convolution. Electronics, 11(18), 2973. https://doi.org/10.3390/electronics11182973
6. Li, R., Wang, S., Zhu, F., & Huang, J. (2018). Adaptive Graph Convolutional Neural Networks. Proceedings of the AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence, 32(1). https://doi.org/10.1609/aaai.v32i1.11691
7. Xu, W., Wu, M., Zhu, J., & Zhao, M. (2021). Multi-scale skeleton adaptive weighted GCN for skeleton-based human action recognition in IoT. Applied Soft Computing, 104(107236), 107236. https://doi.org/10.1016/j.asoc.2021.107236
8. Keskes, O., & Noumeir, R. (2021). Vision-based fall detection using ST-GCN. IEEE Access: Practical Innovations, Open Solutions, 9, 28224–28236. https://doi.org/10.1109/access.2021.3058219
9. Kopach, O. (2020, March 2). AI fall detection module for video analytics platform. Abto Software. https://www.abtosoftware.com/portfolio/researchengineering/fall-detection-for-video-analytics
10. Wizard, H. [@HalflingWizard]. (2021, May 30). Attention Mechanism In a nutshell. Youtube. https://www.youtube.com/watch?v=oMeIDqRguLY
11. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., & Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. https://doi.org/10.48550/ARXIV.1502.03044
12. Yang, X. (2020). An overview of the attention mechanisms in computer vision. Journal of Physics. Conference Series, 1693(1), 012173. https://doi.org/10.1088/1742-6596/1693/1/012173

13. Mnih, V., Heess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. https://doi.org/10.48550/ARXIV.1406.6247
14. “Soft & hard attention.” (n.d.). Github.Io. Retrieved February 20, 2023, from https://jhui.github.io/2017/03/15/Soft-and-hard-attention/
15. Guo, M.-H., Xu, T.-X., Liu, J.-J., Liu, Z.-N., Jiang, P.-T., Mu, T.-J., Zhang, S.-H., Martin, R. R., Cheng, M.-M., & Hu, S.-M. (2021). Attention Mechanisms in Computer Vision: A Survey. https://doi.org/10.48550/ARXIV.2111.07624
16. Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2017). Squeeze-and-Excitation Networks. https://doi.org/10.48550/ARXIV.1709.01507
17. Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). CBAM: Convolutional Block Attention Module. https://doi.org/10.48550/ARXIV.1807.06521
18. Song, S., Lan, C., Xing, J., Zeng, W., & Liu, J. (2016). An end-to-end spatio-temporal attention model for human action recognition from skeleton data. https://doi.org/10.48550/ARXIV.1611.06067
19. Kim, J., Li, G., Yun, I., Jung, C., & Kim, J. (2021). Weakly-supervised temporal attention 3D network for human action recognition. Pattern Recognition, 119(108068), 108068. https://doi.org/10.1016/j.patcog.2021.108068
20. Enea Cippitelli, Ennio Gambi, Samuele Gasparrini, Susanna Spinsante. (2016). TST Fall detection dataset v2. IEEE Dataport. https://dx.doi.org/10.21227/H2QP48
21. Alzahrani, M. S., Jarraya, S. K., Salamah, M. A., & Ben-Abdallah, H. (2017). FallFree: Multiple fall scenario dataset of cane users for monitoring applications using Kinect. 2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), 327–333.
22. Shahroudy, A., Liu, J., Ng, T.-T., & Wang, G. (2016). NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. https://doi.org/10.48550/ARXIV.1604.02808
23. Wang, I. (n.d.). KinectXEFTools: Tools for extracting/converting Kinect Studio XEF files on multiple platforms.
24. Min, W., Yao, L., Lin, Z., & Liu, L. (2018). Support vector machine approach to fall recognition based on simplified expression of human skeleton action and fast detection of start key frame using torso angle. IET Computer Vision, 12(8), 1133–1140. https://doi.org/10.1049/iet-cvi.2018.5324
25. Yao, L. Y., Yang, W., & Huang, W. (2019). ‘An improved feature-based method for fall detection. Gazette, 26(5), 1363–1368.
26. Seredin, O. S., Kopylov, A. V., Huang, S.-C., & Rodionov, D. S. (2019). A skeleton features-based fall detection using Microsoft Kinect v2 with one class-classifier outlier removal. ISPRS - International Archives of the Photogrammetry Remote Sensing and Spatial Information Sciences, XLII-2/W12, 189–195. https://doi.org/10.5194/isprs-archives-xlii-2-w12-189-2019

27. Alzahrani, M. S., Jarraya, S. K., Ben-Abdallah, H., & Ali, M. S. (2019). Comprehensive evaluation of skeleton features-based fall detection from Microsoft Kinect v2. Signal, Image and Video Processing, 13(7), 1431–1439. https://doi.org/10.1007/s11760-019-01490-9