Apr 14th, 12:00 AM - 12:00 AM

Senior Subject Gait Analysis Using Self-supervised Method

Haben Yhdego
Old Dominion University

Michel Audette
Old Dominion University

Senior Subject Gait Analysis using Self-supervised Method

Haben Yhdego
Michel Audette

Department of Computational Modeling and Simulation Engineering
Old Dominion University
{hyhde001, maudette}@odu.edu

Abstract

There are many issues regarding the technology used for fall detection gait analysis in the geriatric center of senior patients. The fall detection system used should examine the privacy of the patients and the flexibility of the system. Several researchers have developed fall detection using wearable sensors due to their flexibility and nature of privacy. Most of those developed methods are supervised deep learning methods. However, data annotation is expensive because we use camera video recording and playback of each participant's recorded video to label the data. Moreover, labelling using a camera recording limits the flexible and private nature of wearable sensor-based fall detection. This paper presents how to use unlabelled data to pre-train our models and use labelled data to fine-tune those pre-trained weights. We collected unlabelled and labelled data and applied self-supervised learning to detect falls. First, we performed pre-training on the unlabeled data using ResNet model. After that, fine-tune and train ResNet using the labelled dataset. The experiment in this study suggested that the best performance can be achieved by using pre-trained weights of unlabelled data from the accelerometer and gyroscope sensors. Furthermore, oversampling and modified loss functions are used to handle the dataset's imbalance classes. With the ResNet pre-trained weights and re-training using the labelled data, the experiments achieved an F1-Score of 0.98.

Keywords: Gait Analysis, Fall Detection, Senior Subjects, Self-supervised Learning.

1 Introduction

The population of senior adults older than 65 years old is growing all over the world, and the US seniors have shown rapid growth in population since 1950 (Statista 2021). In 2019, there was approximately 16 percent of adults aged 65 or over old, and this is expected to reach 22 percent in the next thirty years (Statista 2021). One-third of the ageing population(age 65) fall each year, and all senior adults above 80 years fall annually. USA statistics show that about 36 million falls happen annually, and one-fourth (28%) of the elders above 65 years old, the report falls yearly (Moreland et al. 2020). Of those falls, 37 percent of them (8 million) were injured and needed medical treatment (Moreland et al. 2020). Due to these reasons, there is a national imperative to develop a cost-effective real-time fall detection underpinned by new sensor technologies and methods that are less expensive to develop.

Many researchers has developed classical machine learning based fall detection systems (Ramachandran and Karuppiah 2020). Different classifiers methods are used with hand-crafted feature extractions for real time fall detection systems, such as logistic regression (Putra et al. 2017), Naïve Bayes (Liu et al. 2018)(Putra et al. 2017), decision tree (Putra et al. 2017) (Liu et al. 2018), support vector machines (Putra et al. 2017), and k-nearest neighbors (Medrano et al. 2014)(Putra et al. 2017) (Liu et al. 2018). We need enough labelled data to train the supervised deep learning methods that the above researchers propose.

Last year Meta's Yann LeCun proposed self-supervised learning (LeCun 2021). Self-supervised learning obtains supervisory signals from the data itself, often leveraging the underlying structure in the data. The general technique of self-supervised learning is to predict any unobserved or hidden part (or property) of the input from any observed or unhidden part of the input (LeCun 2021). Self-supervised learning can learn complex patterns using unlabelled data, achieving many state-of-the-art results in different applications. Inspired by the success of BERT: pre-training(self-supervised) for language understanding (Devlin et al. 2018) in natural language processing (NLP), this paper explores pre-trained-based fall detection systems on wearable sensor datasets. A residual neural network (ResNet) (He et al. 2016) is used as a baseline model. For balancing the data sampling of the minor classes, we use oversampling methods (duplicate the minor classes).

2 Materials and Methods

We use Deep Residual Network (ResNet) for pre-training and fine-tuning wearable sensor signals. First, we discuss about the data pre-processing and slide windowing steps and then explain the pre-training and fine-tuning of the baseline deep learning model.

2.1 Data Pre-processing

First, the data points of each feature are normalized (FeatNorm) using the maximum and minimum value of the features, as shown in the equation below. Most real-time fall detection applications should respond within less than 1 second. Therefore, we segment into a 1-second (100 sample signal) for each label (labelled data only) using a fixed-size overlapping sliding window. The size window overlapping (stride) is 0.5 seconds. If we got a single row of falls in this 1-second window signal, this window is labelled fall.

$$FeatNorm = \frac{Feat - min(Feat)}{max(Feat) - min(Feat)} \tag{1}$$

The labelled dataset collected has 13% of the fall dataset and 87% of non-fall activities; it is highly imbalanced data. The main problem of not considering such imbalanced datasets is that our deep learning models

make our minor label classes suffer from low results. However, the accuracy of those minority classes is the most important one. Common approaches to resolving this problem are data-centric and model-centric. The most common and straightforward data-centric sampling methods are random over-sampling, which duplicates random sequences of window signals from the minority class in the training datasets and random under-sampling, which remove the random sample signals from the majority class. We use over-sampling rather than under-sampling because under-sampling for the majority class loses some information, whereas oversampling for the minority class does not lose any data. We use a modified loss function- a weighted focal loss for the model-centric.

2.2 Baseline Models

We ResNet deep learning models as base models for pre-training and fine-tuning our dataset.

Deep Residual Network (ResNet)
The second baseline model used for our self-supervised learning is called deep Residual Network (ResNet) (Wang et al. 2017). The ResNet model has eleven layers- of these layers nine of them are convolution followed by global average pooling that calculates the average of the input sensor signal dimension. The main difference of ResNet from other convolution-based deep learning models are they introduce a residual connection between successive convolutional layers. In addition to that, the ResNet makes the deep learning training fast by decreasing the vanishing gradient problem. ResNet achieves this fast training by using a linear shortcut between the successive residual blocks that make the flow of the gradient directly through these residual connections (He et al. 2016). The ResNet model has three stacked residual blocks and each residual block has three convolutions. The result of each residual block is added to the input of each residual block to be inputted to the next layer. The output of the last residual block is then followed by a global average pooling layer and then a softmax classifier with two neurons as our number of classes are two (fall and non-fall). In all the residual blocks, the three convolutions have kernel lengths of 8,8, and 5 with 64 filters for three of the convolution. The convolution layers are followed by batch normalization and ReLu operations.

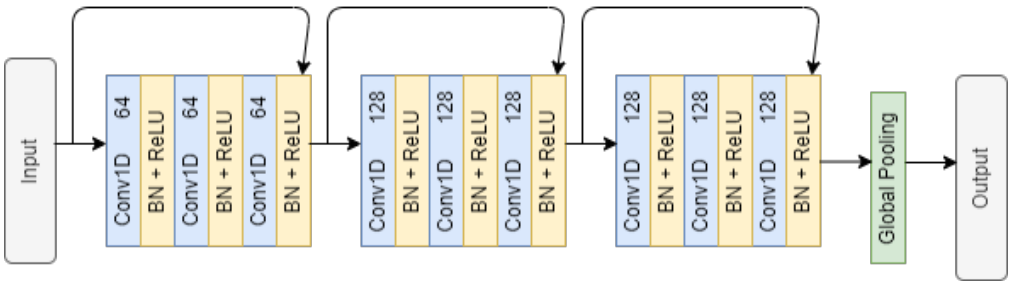

Figure 1: ResNet model architecture (Wang et al. 2017) with respective filter sizes to be used for pre-training and fine-tuning

2.3 Pre-Training (Training on unlabeled data)

Transfer learning from a pre-trained model is a common type of self-supervised learning-based method. Self-supervised learning aims to extract the useful underlying representation of unlabeled data, and the learned data representations can be transferred to downstream tasks. In this way, the problem of label data shortage can be solved. With our work, we exploit ResNet base model, which solve the self-supervised task

by forcing the models to learn filters to solve the gait analysis of fall and non-fall activities. The pre-trained model will be automatically saved and fine-tuned or trained in the same architecture using datasets with labels when training is finished.

We use pre-trained weights of unlabelled datasets to establish a self-supervised critical feature extraction method that helps us get the best results. This process avoids recording the subjects using cameras for manual annotations and does not require any prior knowledge. We obtain pre-training data from a large volume of unlabelled fall detection datasets. We get pre-training weights for fine-tuning with small labelled data and train a linear classifier on top of the model's trained layers. The details of pre-training are as shown in Algorithm 1.

Algorithm 1: Function [P]=Pre-training

input : Unlabelled $acc(t) = [acc_x(t), acc_y(t), acc_z(t)]$ and $gyro(t) = [gyro_x(t), gyro_y(t), gyro_z(t)]$
output: Pre-trained ResNet model weights

for *all data sequence activities of the dataset* **do**
    Normalize all feature columns using equation 4;
    Perform overlapping slide window with 100 samples window size and 50 samples stride ;
Transform the input data in to 3D format of Keras(num of samples * num of features * num steps)
Pre-train the unlabelled data using the two models.
Save the pre-trained weights

2.4 Fine-Tuning and Training (Training on labeled data)

In this case, there are two options for using the pre-trained weights: fine-tuning and training. In fine-tuning, we train the last layer of the model, freeze the other layers, and train for half of the epochs used during pre-training (50/100 epochs). However, we train the whole network in "training" as we did the same strategy during pre-training, except we use the labelled data.

Algorithm 2: Function [F]=Fine-tuning or re-training

input : Pre-trained weights and Labelled $acc(t) = [acc_x(t), acc_y(t), acc_z(t)]$ and
$gyro(t) = [gyro_x(t), gyro_y(t), gyro_z(t)]$
output: Classes of Fall and ADL

for *all data sequence activities of the dataset* **do**
    Normalize all feature columns using equation 4;
    Perform overlapping slide window with 100 samples window size and 50 samples stride ;
    if *there is a single sample out of the 100 samples labelled as fall* **then**
        Label the whole observation window as Fall;
    else
        Label ADL;
Perform over-sampling by duplicating the minor classes
Transform the input data in to 3D format of Keras(num of samples * num of features * num steps)
Apply the weighted focal loss function of equation 2
Classify using Sigmoid classifier
Get the classes of Fall and ADL

Using the labelled dataset, we test both training and fine-tuning using the algorithm shown in 2. To train or fine-tune the labelled datasets, we modify the focal loss presented by (Lin, Tsung-Yi and Goyal, Priya and Girshick, Ross and He, Kaiming and Dollár, Piotr 2017). During supervised training of the labelled dataset, ResNet network are optimized end-to-end using a modified weighted focal loss in Eq. 2:

$$L_{FocalLoss} = \sum_{i=1}^{c=2} w_i (1-p_i)^{\gamma} \log(p_i), \tag{2}$$

where

$$w_i = \frac{n_0 + n_1}{2 * n_i}$$

and n_0 is number of non-fall class, n_1 is number of fall class, and γ is a focusing parameter who value is $\gamma >= 0$. γ is the focusing parameter that specifies to reduce the influence of higher-confidence classified samples in the loss. The higher the , the higher the rate at which easy-to-classify examples are down-weighted. If $\gamma = 0$, weighted focal loss is equivalent to weighted binary cross entropy loss.

3 Results and Discussion

We use balanced accuracy and F1 score performance metrics for comparing the different methods. Balanced accuracy is raw accuracy where each sample is weighted according to the inverse prevalence of its actual class, and it is calculated by the average of recall obtained on each class. The balanced accuracy helps us deal with imbalanced datasets by avoiding inflated performance estimates on the datasets. If the classifier performs equally well on either class, this term reduces to the standard accuracy. In contrast, if the standard accuracy is above chance only because the classifier takes advantage of an imbalanced test set, then the balanced accuracy, as appropriate, will drop to $\frac{1}{n\_classes}$.

$$F1\ Score = \frac{2 * (precision * recall)}{precision + recall} \tag{3}$$

$$Balanced\ Accuracy = \frac{specificity + sensitivity}{2}, \tag{4}$$

where $Precision = \frac{TP}{TP+FP}$, $Specificity = \frac{TN}{TN+FP}$ and $Sensitivity(Recall) = \frac{TP}{TP+FN}$. In addition, the experiments have been implemented using the Keras framework and the labelled datasets used for re-training and fine-tuning are divided into training, validation and testing with 50%, 20%, and 30% respectively. We run the training twenty times to get the average results with a learning rate of 0.001, a batch size of 64, and 100 epochs.

This paper evaluated self-supervised learning for fall detection based on acceleration and angular velocity sensors. Random over-sampling for balancing the fall dataset is used. The performance of those different classifiers is shown in table 1, the self-supervised methods performed better than supervised learning.

Regarding the self-supervised methods, training the pre-trained baseline models from scratch provides better performance in most of the results than fine-tuning the pre-trained baseline model. Even though random oversampling does not add new datasets as it simply duplicates some of the examples, the results are slightly better. When considering the obtained performance results, as we can see in table 1, over-sampling with training the pre-trained ResNet baseline model outperformed other combinations of methods in both of the performance metrics- F1-score and balanced accuracy.

Table 1: Comparing the different methods of ResNet models for fall detection.

Metrics	Supervised ResNet	Self-supervised ResNet	Self-supervised ResNet + Over-sampling
Balanced Accuracy	0.91	0.98	**0.99**
F1-score	0.83	0.96	**0.98**

4 Conclusion

This work proposed a fall detection study using self-supervised learning that pre-trains unlabeled data and fine-tunes using small labelled data. We use overlapping sliding windows for feature extraction and modified weighted focal loss function and data augmentation methods for balancing class data samples. The proposed ResNet self-supervised deep learning method with over-sampling identified the falls against the non-fall activity with an average F-1 score of 0.98. The performance clearly shows that our proposed approach improves the results by using pre-trained models and modified loss function with over-sampling for balancing the datasets.

We examine that ResNet learned weights have a strong correlation due to an over-parameterized model. In our future work, we are planning to use de-correlation of filters regularization for both networks. We are going to calculate the total loss which is the summation of the weighted focal loss and de-correlation loss. Moreover, we will further explore and improve the experimental data, model architecture, and hyper-parameters.

REFERENCES

Devlin, J., M. Chang, K. Lee, and K. Toutanova. 2018. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". *CoRR* vol. abs/1810.04805.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep residual learning for image recognition". *IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Yann LeCun 2021. "Self-supervised learning: The dark matter of intelligence".

Lin, Tsung-Yi and Goyal, Priya and Girshick, Ross and He, Kaiming and Dollár, Piotr 2017. "Focal Loss for Dense Object Detection".

Liu, K., C. Hsieh, S. Hsu, and C. Chan. 2018. "Impact of Sampling Rate on Wearable-Based Fall Detection Systems Based on Machine Learning Models". *IEEE Sens. J*, pp. 9882–9890.

Medrano, C., R. Igual, I. Plaza, and M. Castro. 2014. "Detecting falls as novelties in acceleration patterns acquired with smartphones.". *PLoS ONE*, pp. 9.

Moreland, B., R. Kakara, and A. Henry. 2020. "Trends in Nonfatal Falls and Fall-Related Injuries Among Adults Aged 65 Years — United States". *MWR Morb Mortal Wkly Rep* vol. 69, pp. 875–881.

Putra, I., J. Brusey, E. Gaura, and R. Vesilo. 2017. "An Event-Triggered Machine Learning Approach for Accelerometer-Based Fall Detection". *Sensors*, pp. 20.

Ramachandran, A., and A. Karuppiah. 2020. "A Survey on Recent Advances in Wearable Fall Detection Systems". *Biomed Res. Int.*, pp. 365–368.

Statista 2021. "USA seniors as a percentage of the population 1950-2050".

Wang, Z., W. Yan, and T. Oates. 2017. "Time series classification from scratch with deep neural networks: a strong baseline". *International joint conference on neural networks*, pp. 1578–1585.