FALL DETECTION USING SELF-SUPERVISED PRE-TRAINING MODEL

Haben Yhdego
Michel Audette

Modeling and Simulation Engineering
Old Dominion University
5115 Hampton Blvd
Norfolk, VA, USA
{hyhde001, maudette}@odu.edu

Christopher Paolini

Electrical and Computer Engineering
San Diego State University
5500 Campanile Drive
San Diego, CA, USA
cpaolini@sdsu.edu

ABSTRACT

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. This paper presents how to use unlabeled data to pre-train FCN and ResNet models, and use labeled data to fine-tune those pre-trained weights. We collected unlabeled and labeled data and applied self-supervised learning to detect falls. The experiment in this study suggested that the best performance can be achieved by using pre-trained weights of unlabeled 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 training using the labeled data, the experiments achieved an F1-Score of 0.98.

Keywords: Fall Detection, Wearable Sensors, Self-supervised Learning, Imbalanced Dataset.

1 INTRODUCTION

The population of senior adults older than 65 years old is growing worldwide, and the US seniors have shown rapid growth in population since 1950 (Statista 2021). In 2019, approximately 16 percent of adults aged 65 or over are expected to reach 22 percent in the next thirty years (Statista 2021). One-third of the aging population (age 65) falls 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 falls yearly (Moreland et al. 2020). Of those falls, 37 percent (8 million) were injured and needed medical treatment (Moreland et al. 2020). Furthermore, wearing a fall alert system helps to alleviate a fear of falling. Not fearing to fall, seniors don't curtail their activities, allowing them to remain physically active and preventing functional decline. It can also help other pre-defined anomalous or predictive behavior and can alert third parties for intervention in advance or after a fall, thus remediating impact (Yhdego et al. 2021). Due to these reasons, a national imperative is to develop a cost-effective real-time fall detection underpinned by new sensor technologies and methods.

There are many issues regarding the technology used for fall detection systems 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. Wearable sensor device with inertial sensors of accelerometer and gyroscope used as the primary data source helps us avoid those issues. The analysis of the signals obtained from human body-mounted wearable sensors are commonly used to monitor the health status of older patients with movement assistive devices (Bright and Coventry 2013). These sensors usually generate complex hip motion signals, which are difficult to interpret without expert intervention. A computationally efficient fall detection modelling technique that will provide a meaningful characterization of the sensor data is required to automatically analyze the sensor readings to infer the kind of human activity performed by a user. Many researchers have been developing supervised-based fall detection methods in the last decades. However, there are still some limitations to the manual annotation of the dataset collected using wearable sensors.

Meta's Yann LeCun proposed self-supervised learning (Yann LeCun 2021). Self-supervised learning obtains supervisory signals from its dataset, 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 (Yann LeCun 2021). Self-supervised learning can learn complex patterns using unlabeled 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. 2019) in natural language processing (NLP), this paper explores pre-trained-based fall detection systems on wearable sensor datasets. Two different deep learning models are used as a baseline model-fully connected networks (FCN) (Wang et al. 2017) and a residual neural network (ResNet) (He et al. 2016). To balance the minor classes' data sampling, we use random oversampling methods.

2 RELATED WORKS

Deep learning algorithms have been applied to several areas, such as computer vision, image recognition, human activity recognition, and fall detection systems. Fall detection system development mainly uses simulated fall data sets by well-protected non-patient subjects to be then used and validated by real patients (Bagalà et al. 2012, Klenk et al. 2011). A fall detection system based on simulated falls of such young subjects was developed by Chaudhuri et al. (2015). Other fall detection devices based on real patients are developed by Lipsitz et al. (2016), but the false positive and false negative are high.

Many researchers have 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), Naive 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).

Recurrent neural network (RNN) and long short-term memory (LSTM) are popular algorithms used in sequence models to encode and process sequential data. As a wearable sensor-based fall detection method generates sequential datasets, it has the advantage of using these sequential models. Sensor fusion of accelerometer and gyroscope data streams using a hybrid CNN-LSTM method proposed by Delgado-Escaño et al. (2020), and LSTM-based activity recognition by Paolini et al. (2019) and Aicha et al. (2018), are used for fall recognition. The study by Aicha et al. (2018) employs a single inertial sensor placed on the trunk. The CNN and LSTM model inputs are the raw acceleration and angular velocity signals. The problem with Ruben's approach is that he uses KNN classifiers- which it is challenging to use such algorithms for real-time fall detection due to the computational time of KNN classifiers. We need enough labeled data to train the supervised deep learning methods proposed above.

BERT is a pre-training learning model that obtains state-of-the-art results in various NLP tasks (Chen, Zhuo, and Wang 2019, Vig and Ramea 2019). Our approach is based on learning language representation self-supervised BERT (Devlin et al. 2019). We collected unlabeled and labeled fall datasets and pre-processed and balanced the minor class data samples. Fall events rarely occur in relation to other daily activities, so the dataset collected from actual patients is imbalanced. Using imbalanced data for training deep learning models fail when we try to test on the minority class of imbalanced datasets, which happens in actual patient fall events. Thus, an essential requirement of the analysis is the consideration of methodologies for compensating for an imbalanced dataset in deep learning methods. We use modified weighted focal loss and random oversampling to handle the dataset's imbalanced nature.

3 MATERIALS AND METHODS

We use two base models-Fully Convolutional Neural Networks (FCNs) and Deep Residual Network (ResNet) for pre-training and fine-tuning wearable sensor signals. First, we discussed the labeled and unlabeled datasets we collected and then explained the required pre-processing and slide windowing steps. Lastly, the pre-training and fine-tuning of the baseline deep learning models are explained.

3.1 Data

The human subjects' fall experiment is carried out with safety measures at the neuromechanics and neuroplasticity lab of San Diego State University by Dr Paolini. Each subject walked straight (back and forth) by wearing the VR headset to a path in the laboratory. The VR headset would depict a straight sidewalk for the participant with no obstacles or deviations in the middle of the track, ensuring that the participant is familiar with his virtual environment while walking. Mattresses were placed at the front and along the side of the path to prevent injuries. Up on falling, the sensors are checked for dislocation and adjusted accordingly before performing the next fall experiment. The collected fall dataset has a sampling rate of 100 HZ, mounted in the shank (Shinbone) (Noraxon 2019). The lab was set up with wireless 3D Motion capture cameras which record the human subject movements (San Diego State University 2018) for annotating the labeled data only. The data were collected from sixteen subjects between the ages of 20-50; two were females, and the rest were males.

Many fall detection studies (Casilari et al. 2020) utilize an accelerometer as a primary sensor to determine falls. However, using only the acceleration measurements can result in many false positives and false negatives caused by near-fall activities such as sitting down fast on a mattress. Near fall and fall activities have almost the same vertical signal variation, making it difficult to differentiate. False-positive and false negatives caused by the near-fall activities can be reduced significantly using the gyroscope's angular velocity measurements. Hence, to detect falls with low energy consumption, we use the acceleration along the x, y, and z-axis and angular velocity along the x, y, and z-axis. The Noraxon myoMotion research inertial measurement unit (IMU) sensors are used to measure those features. These feature values were saved as comma-separated value (CSV) files accessed using the Noraxon MR3 Software and exported to the computer (Paolini et al. 2019). A clean fall is considered a proper fall event, labelled with a binary value '1' in each test case. At the same time, non-fall activities (standing, walking) and near-fall activities (slipping, stumbling, and sitting) are non-fall events labeled with a binary value '0'. Our deep learning models use these CSV files of different human subjects as train and test datasets.

3.2 Data Pre-processing

First, the data points of each feature (Feat)- x, y, and z of the accelerometry and gyroscope, are normalized (Feat_Norm) using the maximum and minimum values 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 (labeled 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

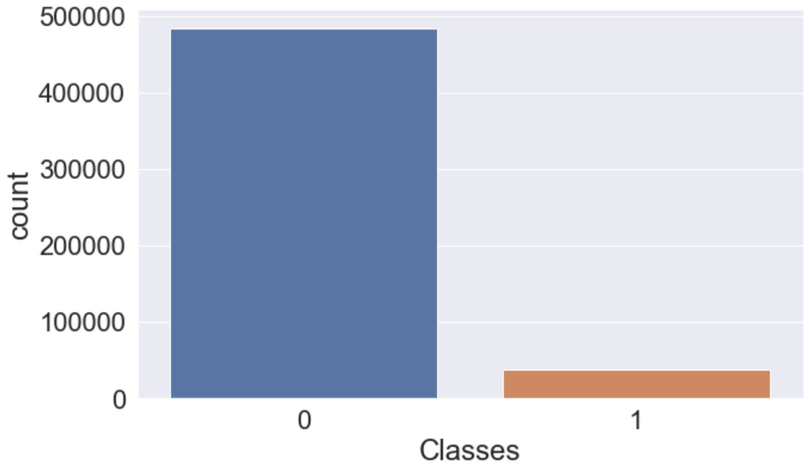

Figure 1: The class proportion of our labeled dataset. Number of rows for non-fall activity (class 0) and fall activity (class 1)

window signal, this window is labeled fall.

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

The dataset contains 30% of labeled dataset and 70% of unlabeled dataset. The labeled dataset collected has 13% of the fall dataset and 87% of non-fall (ADL) activities; it is highly imbalanced data, as shown in Figure 1. 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 randomly selects examples from the minority class with replacement and adds them to the training dataset and random under-sampling, which removes the random sample signals from the majority class. We use random 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.

3.3 Baseline Models

Self-supervised learning is a good technique for learning features in the absence of enough labeled datasets. It is very efficient where labeling data is expensive, as in our case, we need to install a camera for manually annotating. In this work, we propose a self-supervised fall detection model, as shown in Figure 2 that extracts representative features learned from unlabeled data. During pre-training, the base models leverage features learned on unlabeled data. After we pre-trained, we will fine-tune the learned weights to the labeled fall dataset. We use FCN and ResNet deep learning models as base models for pre-training and fine-tuning our dataset.

Fully Convolutional Networks (FCN)

Fully Convolutional Neural Networks (FCNs) have proven to be an effective learning model for sensor signal sequence data (Wang et al. 2017). FCNs are the same as the standard convolutional neural networks (CNN) without the local pooling layers - the sensor signal's input size will not change when it goes further into the deep layers of the convolutions (Wang et al. 2017). Moreover, the main difference between CNN

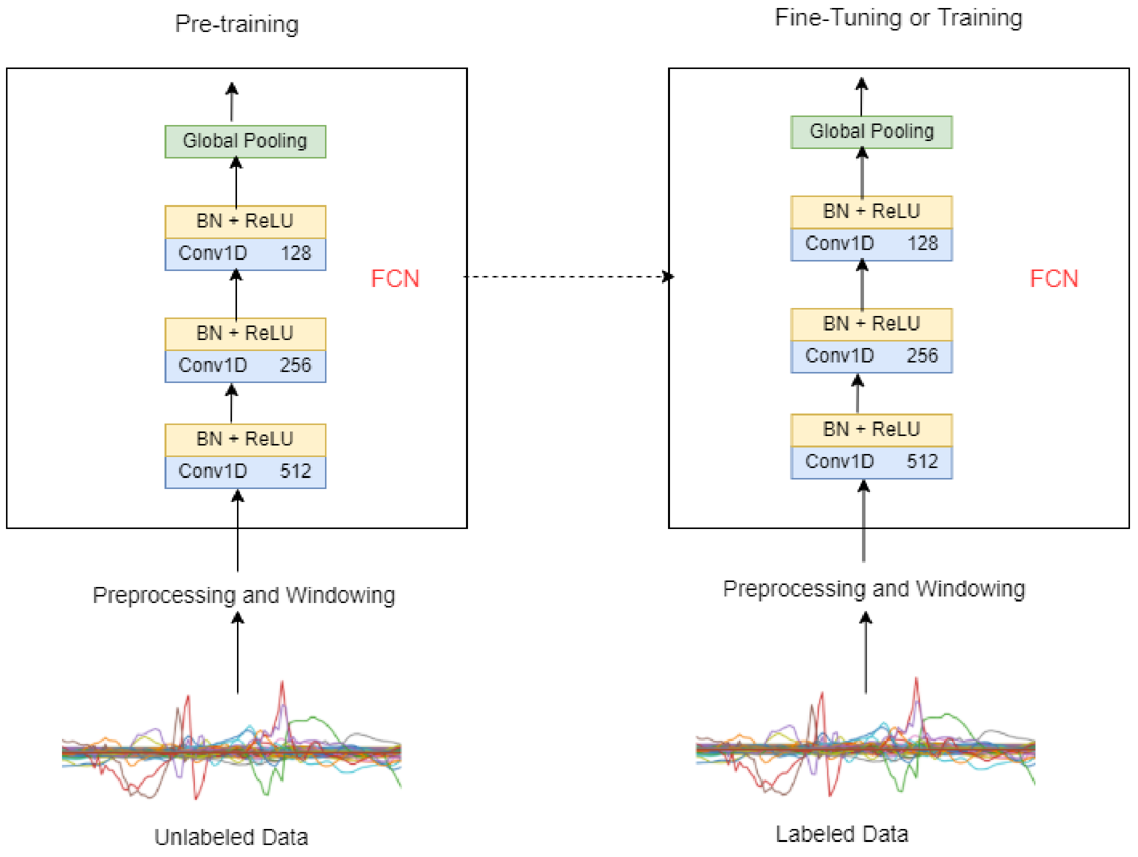

Figure 2: The self-supervised architecture

and FCNs models is that the last fully connected layer is replaced by global average pooling in FCNs one of the main characteristics of this architecture is the replacement of the traditional final fully-connected layer. Hence, avoiding the fully connected layer helps us decrease the number of parameters learned in the neural network. Our model adapted from (Wang et al. 2017) as shown in Figure 2 consist of three stacked convolution layers of block where a single block has a convolution, batch normalization (Ioffe and Szegedy 2015), and ReLU sub-layers. As we can see from the model, there are no sub-sampling (pooling) and dropout (regularization) layers. The three convolution layers have 512 filters with a kernel size of 5, 256 filters with a kernel size of 5, and 128 filters with a kernel size of 3 consecutively. Each convolution layers use a stride length of 1 with zero paddings to keep the sequence length of the input signal after the convolution operation. Each convolution layer is followed by batch normalization and then passes through the ReLU activation function. The global average pooling layer was applied to the result found from the last convolution. Finally, a sigmoid classifier is used for classifying falls from non-fall activities.

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-nine of which are convolution, followed by global average pooling that calculates the average of the input sensor signal dimension. The main difference between ResNet and other convolution-based deep learning models is that they introduce a residual connection between successive convolutional layers. And, 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 direct the gradient flow 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. The output of the last residual block is then followed by a global average pooling layer and then a sigmoid classifier. In all the residual blocks, the three convolutions have kernel lengths of 8, 8, and 5 with 64, 128, and 128 filters for three of the convolution as shown in Figure 3. The convolution layers are followed by batch normalization and ReLu operations. All the layers have an invariant number of parameters, similar to FCN.

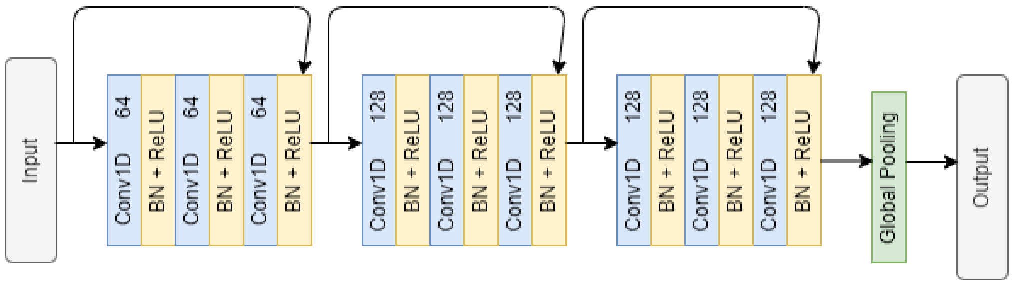

Figure 3: ResNet model architecture (Wang et al. 2017) with respective filter sizes to be used for pre-training and fine-tuning by replacing FCN in the framework above.

3.4 Pre-Training (Training on unlabeled data)

Transfer learning from a pre-trained model is one type of self-supervised learning method. Self-supervised learning aims to extract the useful underlying representation of unlabeled data and transfer these learned data representations to downstream tasks. In this way, it can solve the problem of labeled data shortage. Our work exploits FCN, ResNet base models, which solve the self-supervised task by forcing the models to learn filters to solve the gait analysis of fall and non-fall activities. To get the pre-trained weights of the unlabeled data, we use mixup data augmentation (Zhang et al. 2018) for generating synthetic data and calculate the cosine similarity between the original and generated data. When training is finished, the pre-trained weights will be automatically saved and fine-tuned or trained in the same architecture using datasets with labels.

We use pre-trained weights of unlabeled fall datasets to establish a self-supervised critical feature extraction method that helps us get better results. This process avoids recording the subjects using cameras for manual annotations and does not require any prior knowledge. The pre-training weights are fine-tuned with small labeled data and trained with 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  : Unlabeled $acc(t) = [acc_x(t), acc_y(t), acc_z(t)]$ and $gyro(t) = [gyro_x(t), gyro_y(t), gyro_z(t)]$
output: Pre-trained FCN/ResNet model weights

for all data sequence activities of the dataset do
    Normalize all feature columns using equation 1;
    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)
Perform data augmentation using Mixup (Zhang et al. 2018).
Train the unlabeled data using cosine similarity.
Save the pre-trained weights
```

3.5 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 labeled data.

```
Algorithm 2: Function [F]=Fine-tuning or training
input  : Pre-trained weights and Labeled $acc(t) = [acc_x(t), acc_y(t), acc_z(t)]$ and
         $gyro(t) = [gyro_x(t), gyro_y(t), gyro_z(t)]$
output: Classes of Fall and ADL (Non-Fall)
for all data sequence activities of the dataset do
    Normalize all feature columns using equation 1;
    Perform overlapping slide window with 100 samples window size and 50 samples stride ;
    if there is a single sample out of the 100 samples labeled as fall then
        Label the whole observation window as Fall;
    else
        Label ADL (Non-Fall);
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 (Non-Fall)
```

Using the labeled dataset, we test training and fine-tuning using the method shown in Algorithm 2. To re-train or fine-tune the labeled datasets, we modify the focal loss presented by (Lin et al. 2020). During supervised training of the labeled dataset, FCN and ResNet networks 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 whose value is $\gamma >= 0$. This focusing parameter specifies to reduce the influence of higher-confidence classified samples in the loss. The higher the γ, the higher the rate at which easy-to-classify examples are down-weighted. If $\gamma = 0$, a weighted focal loss is equivalent to a weighted binary cross-entropy loss.

We use balanced accuracy and F1 score performance metrics for comparing the different methods. Balanced accuracy is a raw accuracy, where each sample is weighted according to its actual class's inverse prevalence. It helps us to 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 classical accuracy is above chance only because the classifier takes advantage of an imbalanced test set, then the balanced accuracy, as appropriate, will drop to $\frac{1}{n\_classes}$.

$$F1score = \frac{2 * (precision * recall)}{precision + recall}$$

$$Balanced accuracy = \frac{specificity + sensitivity}{2},$$

where $Precision = \frac{TP}{TP+FP}$, $Specificity = \frac{TN}{TN+FP}$ and $Sensitivity(Recall) = \frac{TP}{TP+FN}$.

4 RESULTS AND DISCUSSION

This paper evaluated self-supervised learning for fall detection based on acceleration and angular velocity sensors. The experiment has been implemented using the Keras framework. The labeled datasets used for training and fine-tuning are divided into training, validation and testing with 50%, 20%, and 30%, respectively. We ran the training twenty times to get the average results with a learning rate of 0.001, a batch size of 64, and 100 epochs. The performance of those different classifiers is shown in Tables 1 and 2. FCN model is slightly better than the ResNet model in supervised learning. This indicates that FCN network size works better than ResNet with small datasets. However, the balanced accuracy and F1 score of ResNet based self-supervised model are higher than those of the FCN self-supervised model. Furthermore, the self-supervised methods performed better than supervised learning for both baseline models- FCN and ResNet.

Table 1: Comparing the different base models of fall detection.

Metrics	Supervised training		Self-supervised			
			Fine-Tuning		Training	
	ResNet	FCN	ResNet	FCN	ResNet	FCN
Balanced Accuracy	0.91	0.93	0.95	0.94	**0.98**	0.96
F1-Score	0.83	0.80	0.86	0.86	**0.96**	0.89

Table 2: Comparing the different methods of ResNet models for fall detection.

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

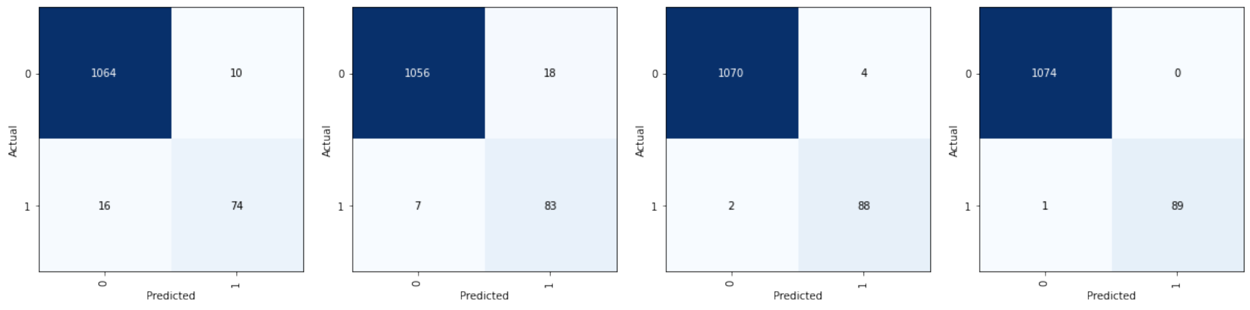

Figure 4: Confusion matrix for the ResNet Model a) Supervised Learning b) Self-supervised (Fine-tuning) c) Self-supervised (Training) d) Self-supervised (Training) + Random Over-sampling

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 random examples, the results are slightly better. ResNet base model outperforms FCN base model in most results as shown in Table 1. Furthermore, random over-sampling with the training of the pre-trained ResNet baseline model outperformed other combinations of methods in both of the performance metrics, as we can see in Table 2 and the confusion matrix in Figure 4. Lastly, we compared the results of the models based on sex characteristics. The performance of ResNet model on the female datasets is almost the same as on the male datasets, with an F1 score of 0.95 and 0.96, respectively.

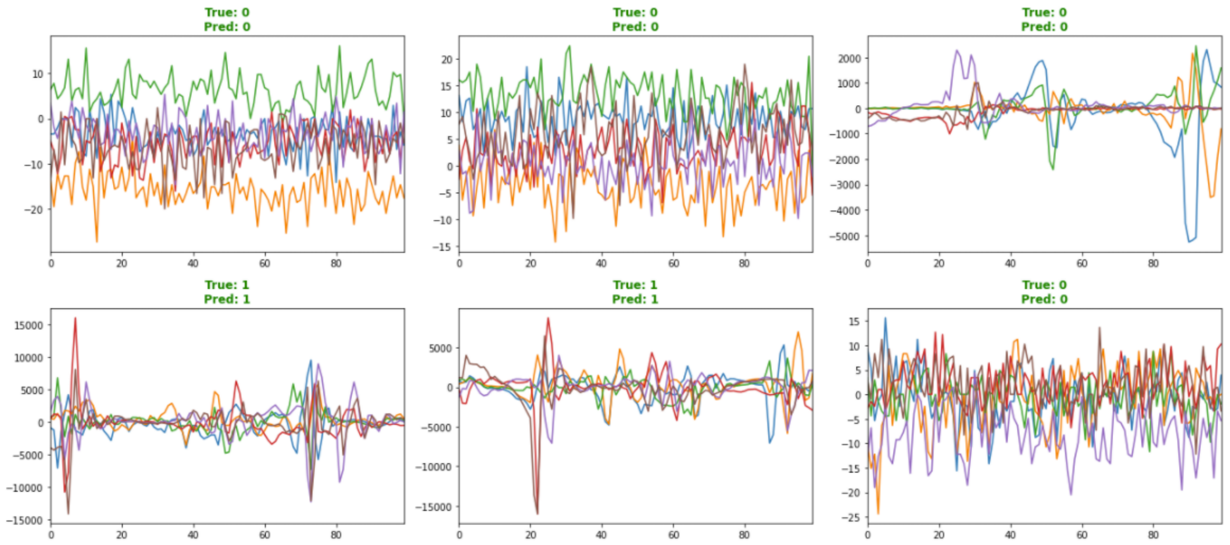

Figure 5: Sample predicted classes using the pre-trained ResNet model. Non-fall activity (class 0), fall activity (class 1), and True, Pred are the actual and predicted classes, respectively.

We try to visualize the actual and predicted classes of the self-supervised ResNet model for sample data points. As shown in Figure 5, the visualization shows the data points, and their corresponding predicted and actual classes. In the third column of the first row, the ADL, a near-fall activity, is correctly classified as non-fall activity. It shows that our self-supervised ResNet model pre-trained using unlabeled datasets is stable even for near-fall events.

5 CONCLUSION

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

Even though we are using a weighted focal loss function for FCN and ResNet, we inspect that ResNet network learned weights strongly correlate due to an over-parameterized network. In our future work, we plan to use the decorrelation of filter regularization for both networks. We will calculate the total loss, which is the summation of the weighted focal loss and decorrelation loss. Besides this, we will further explore and improve the experimental data, model architecture, and hyperparameters.

REFERENCES

Aicha, A. N., G. Englebienne, K. van Schooten, M. Pijnappels, and B. Kröse. 2018. “Deep Learning to Predict Falls in Older Adults Based on Daily-Life Trunk Accelerometry”. *Sensors* vol. 18 (5), pp. 1654.

Bagalà, F., C. Becker, A. Cappello, L. Chiari, K. Aminian, J. M. Hausdorff, W. Zijlstra, and J. Klenk. 2012. "Evaluation of Accelerometer-Based Fall Detection Algorithms on Real-World Falls". *PLoS ONE* vol. 7 (5), pp. e37062.

Bright, A. K., and L. Coventry. 2013. "Assistive Technology for Older Adults: Psychological and Socio-Emotional Design Requirements". In *Proceedings of the 6th International Conference on PErvasive Technologies Related to Assistive Environments - PETRA '13*. Rhodes, Greece.

Casilari, E., M. Álvarez-Marco, and F. García-Lagos. 2020. "A Study of the Use of Gyroscope Measurements in Wearable Fall Detection Systems". *Symmetry* vol. 12 (4), pp. 649.

Chaudhuri, S., D. Oudejans, H. J. Thompson, and G. Demiris. 2015. "Real-World Accuracy and Use of a Wearable Fall Detection Device by Older Adults". *Journal of the American Geriatrics Society* vol. 63 (11), pp. 2415–2416.

Chen, Q., Z. Zhuo, and W. Wang. 2019. "BERT for Joint Intent Classification and Slot Filling". *Computing Research Repository (CoRR)* vol. abs/1902.10909.

Delgado-Escaño, R., F. M. Castro, J. R. Cózar, M. J. Marín-Jiménez, N. Guil, and E. Casilari. 2020. "A Cross-Dataset Deep Learning-Based Classifier for People Fall Detection and Identification". *Computer Methods and Programs in Biomedicine* vol. 184, pp. 105265.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In *NAACL-HLT*. Minneapolis, MN, USA.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep Residual Learning for Image Recognition". In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA.

Ioffe, S., and C. Szegedy. 2015. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In *International Conference on Machine Learning*. Lille, France.

Klenk, J., C. Becker, F. Lieken, S. Nicolai, W. Maetzler, W. Alt, W. Zijlstra, J. Hausdorff, R. van Lummel, L. Chiari, and U. Lindemann. 2011. "Comparison of Acceleration Signals of Simulated and Real-World Backward Falls". *Medical Engineering &amp Physics* vol. 33 (3), pp. 368–373.

Lin, T.-Y., P. Goyal, R. Girshickand, K. He, and P. Dollar. 2020. "Focal Loss for Dense Object Detection". *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 42(2), pp. 318 – 327.

Lipsitz, L. A., A. E. Tchalla, I. Iloputaife, M. Gagnon, K. Dole, Z. Z. Su, and L. Klickstein. 2016. "Evaluation of an Automated Falls Detection Device in Nursing Home Residents". *Journal of the American Geriatrics Society* vol. 64 (2), pp. 365–368.

Liu, K.-C., C.-Y. Hsieh, S. J.-P. Hsu, and C.-T. Chan. 2018. "Impact of Sampling Rate on Wearable-Based Fall Detection Systems Based on Machine Learning Models". *IEEE Sensors Journal* vol. 18 (23), 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* vol. 9 (4), pp. e94811.

Moreland, B., R. Kakara, and A. Henry. 2020. "Trends in Nonfatal Falls and Fall-Related Injuries Among Adults Aged $\geq$65 Years — United States, 2012–2018". *MMWR. Morbidity and Mortality Weekly Report* vol. 69 (27), pp. 875–881.

Noraxon 2019. "MyoRESEARCH 3.14 User Manual". https://www.noraxon.com/. Accessed Feb 23, 2021.

Paolini, C., D. Soselia, H. Baweja, and M. Sarkar. 2019. "Optimal Location for Fall Detection Edge Inferencing". In *2019 IEEE Global Communications Conference (GLOBECOM)*. Waikoloa, HI, USA.

Putra, I., J. Brusey, E. Gaura, and R. Vesilo. 2017. "An Event-Triggered Machine Learning Approach for Accelerometer-Based Fall Detection". *Sensors* vol. 18 (2), pp. 20.

Ramachandran, A., and A. Karuppiah. 2020. “A Survey on Recent Advances in Wearable Fall Detection Systems”. *BioMed Research International* vol. 2020, pp. 1–17.

San Diego State University 2018. “Neuromechanics and Neuroplasticity Laboratory-ENS 216”. https://ens.sdsu.edu/dpt/research/faculty-research-interests/neuromechanics-and-neuroplasticity-lab/. Accessed: Apr. 29, 2020.

Statista 2021. “USA Seniors as a Percentage of the Population 1950-2050”. https://www.statista.com/statistics/457822/share-of-old-age-population-in-the-total-us-population/. Accessed Jun 23, 2021.

Vig, J., and K. Ramea. 2019. “Comparison of Transfer-Learning Approaches for Response Selection in Multi-Turn Conversations”. In *Association for the Advancement of Artificial Intelligence*. Honolulu, HI,USA.

Wang, Z., W. Yan, and T. Oates. 2017. “Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline”. In *2017 International Joint Conference on Neural Networks (IJCNN)*. Anchorage, AK, USA.

Yann LeCun 2021. “Self-Supervised Learning: The Dark Matter of Intelligence”. https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/. Accessed May 23, 2022.

Yhdego, H., J. Li, C. Paolini, and M. Audette. 2021. “Wearable Sensor Gait Analysis of Fall Detection using Attention Network”. In *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. Houston, TX, USA.

Zhang, H., M. Cisse, Y. Dauphin, and D. Lopez-Paz. 2018. “Mixup: Beyond Empirical Risk Minimization”. In *International Conference on Learning Representations (ICLR)*. Vancouver, Canada.

AUTHOR BIOGRAPHIES

HABEN YHDEGO is a PhD student in the Computational Modeling and Simulation Engineering Department at Old Dominion University. He received M.Sc. in image processing and computer vision from Jean-Monet University, France. His research interests include applying machine learning methods for images and sensor datasets, image analysis and computer vision. His email address is hyhde001@odu.edu.

CHRISTOPHER PAOLINI is Assistant Professor in the Department of Electrical and Computer Engineering at San Diego State University. Christopher Paolini’s current research interests include Internet of Things device development, machine learning, embedded systems, deep learning, high performance computing, high speed (100gbps) networking, cyber infrastructure development, and cybersecurity. His email address is cpaolini@sdsu.edu.

MICHEL AUDETTE is an Associate professor of the Department of Computational Modeling and Simulation Engineering at Old Dominion University. He holds a PhD in Biomedical Engineering from McGill University. His research interests include medical simulation, surgery planning and medical image analysis. His email address is maudette@odu.edu.