International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*

Analysis for Symptoms of Human Fall using Pre-Processing and Segmentation based on Deep Learning Architectures

[1]Raj Kumar G, [2]Bevish Jinila Y
[1]Research Scholar, Department of CSE, Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu - 600119, India
[1]annaunivraj@gmail.com
[2]Associate Professor, Department of CSE, Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu - 600119, India
[2]bevishjinila.it@sathyabama.ac.in

Abstract—By building sensor-based alert systems, physical therapists can not only decrease the after-fall repercussions but even save lives.Older people are prone to several diseases, and falling is a regular occurrence for them during this period.Various fall detection systems have recently been developed, with computer vision-based approaches being one of the most promising and effective. Here, the sensor-based data has been analysed for a patient's human fall symptoms. This data has been pre-processed using Gaussian filtering with kernel neural network in which the data has been normalized and trained based on neural network. The trained normalized data has been segmented using encoded Stacked Deconvolutional Network (EnSt-DeConvNet). We found that the suggested method predicts such fall symptoms with the highest accuracy from sensor data. Other algorithms' accuracy results, on the other hand, are also fairly close. Experiments reveal that the suggested technique, when compared to other generally utilized techniques based on multiple cameras fall dataset, produced reliable findings and that our dataset, which consists of more training samples, produced even better results. Experimental results showaccuracy of 96%, Precision of 94%, Recall of 88% and F-1 score of 82%, computational time of 69%.

Keywords- Human fall, computer vision, Gaussian filtering, kernel neural network, EnSt-DeConvNet, accuracy.

I. INTRODUCTION

Due to unintentional falls, health centers are dealing with a large number of patients, resulting in a significant financial burden on society. The typical hospital bill for a fall injury, for example, is more than $30,000. As a result, developing cost-effective solutions to limit injuries caused by falls and providing speedier aid when a fall happens is crucial. Several risk factors for falling, and specialized interventions are created to prevent injuries. Several solutions have been created to this end and are currently available. Most of these methods deal with fall detection [1], and they only alert users' friends when a fall occurs.

However, FPPSs [2] exist to predict and avoid a fall. Without involving users in the monitoring process, such methods track as well as report data through wearable sensors. Sensors collect data and software programs process it: first, data from sensors is collected, and then obtained data is analyzed to extract a suitable feature set. After that, the data is subjected to a machine learning system. Fall risk is often estimated in real-time or the future using fall prediction systems [3].

These devices can help you avoid financial and physical effects of a fall. Because both predictions, as well as prevention methods, assesses fall risk, phrases prediction, as well as prevention, are sometimes used interchangeably. These methods are critical for determining feasibility of implementing recovery measures before a fall.

Because many people require living help, a human fall detection method is significant element of assistive method. There is a notable increase in the senior population in Bangladesh and Western countries. According to statistics on human fall detection, falls are a major cause of injury mortality in persons over the age of 79 [4].According to a study conducted by the National Institutes of Health in United States, approximately 1.6 million senior people in United States are injured by falls each year [5].

Meanwhile, China is undergoing most rapid population ageing in human history, with population estimated to reach approximately 35% by 2050, up from around 30% in 2020.According to a study, 93% of the elderly live alone in their homes, with 29% living alone at a facility. Even if they have no physical injuries, over half of older people laying on the floor for more than an hour owing to a fall will die within six months [6].

The contribution of this paper is as follows:

1) To collect the patient's pre-historic symptoms for fall and monitor sensor data

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*

2) To pre-process the input data in fall symptom analysis using Gaussian filtering with kernel neural network (GF_KNN)
3) To train the normalized processed data for segmentation using encoded Stacked De-convolutional Network (EnSt-DeConvNet)

The experimental results show parametric analysis of proposed fall symptom analysis in terms of accuracy, precision, recall, F-1 score and computational time.

II. RELATED WORKS:

Among the many types of DL methods developed in this decade, CNN has achieved enormous success in image segmentation, object recognition, NLP, machine translation, all of which require a large training dataset to complete [7].According to research by [8,], most falls among the elderly occur in the dark. Fall events in a cluttered indoor environment are classified in [9] by reducing occlusion effects. The ability to recognize a series of stances, on the other hand, is essential for distinguishing non-fall from fall. To distinguish fall frames from non-fall frames, a binary SVM is used. SV Munder performs when the training data sample is insufficient. In [10], a CNN is utilized to distinguish various stances.

Because of shadow, the background removal approach misclassifies some datasets in different illumination situations. As a result, it makes incorrect predictions when bending, crawling, or sitting. The authors of [11, 12] provide a summary of fall detection methods. Both analyses, however, utilize relatively old literature from 2007 and 2008, respectively. Wearable, ambient, and camera-based fall detection techniques were characterized in a study [13].Author [14] discusses the difficulties and trends in fall detection techniques similarly. The research [15] focuses on fall detection with wearable sensors. A review of fall detection, as well as prevention approaches, was presented by the author in [16].A review of gait analysis using ML was recently published in [17].However, just 14% of their research focuses on fall detection and prevention. Work [18] provides an in-depth look into fall detection and prevention practices. However, majority of the methods provided employ statistical methodologies, which frequently result in several false alarms during detection as well as categorization.

Furthermore, in complicated and nonlinear issues, statistical approaches are less effective. Gait analysis for fall detection as well as prevention creates a lot of noisy data throughout the acquisition process. In general, statistical approaches are susceptible to noisy data, which causes performance reduction [19].As a result of the excellent classification accuracy for fall detection as well as prevention, the most recent research adds Machine Learning (ML).A review of fall detection utilizing DL approaches was recently published [20].However, the scope of review is restricted to DL algorithms for fall detection. The characterization of one's quality of movement has recently been shown to have added relevance in determining fall risk in older persons [21].Biomechanical characteristics such as gait stability, variability and smoothness, as well as mean turn time and number of aberrant sit-to-stand transitions, were linked to falling risk in these investigations [22].However, estimating these attributes frequently necessitates event detection, which is necessary for refinement and may not currently take advantage of the quantity of data gathered. DL enables for data-driven construction of features and avoids these drawbacks. Deep convolution and LSTM, RNN have been demonstrated to be effective for recognizing activities as well as gait patterns from inertial sensor data in machine learning [23].

III. SYSTEM MODEL:

This section discusses the proposed analysis of human fall symptoms based on deep learning techniques in processing sensor data. Here the input data has been collected as the patient's pre-historic symptoms for fall and monitored sensor data. This data has been pre-processed and segmented based on neural network techniques in which the data has been normalized and trained in analyzing the symptoms of fall. The pre-processing has been carried out using Gaussian filtering with kernel neural network (GF_KNN) and this normalized trained data has been segmented using encoded Stacked De-convolutional Network (EnSt-DeConvNet). The overall proposed architecture is shown in figure-1.

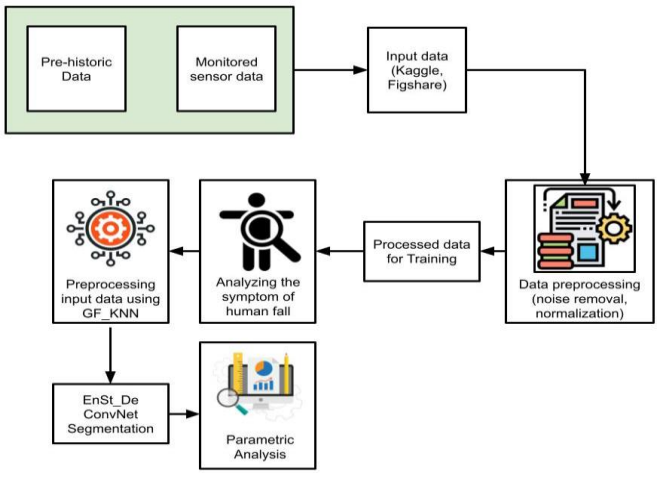


Figure 1. Overall proposed architecture

A. Data Preparation

Python 3.7.3 was used to write the code. As a first step, merge all.txt files for all subjects that were recorded with force-plate. For each 60 second trial, time series were recorded at a frequency of 100 Hz, resulting in 6000 rows. We had 1956.txt files because each of 163 subjects completed all 12 trials. However, for five participants who were unable to finish the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*

most difficult trials, 26 of these.txt files are missing. As a result, there are 11 580 000 rows of data for all subjects. The following attributes make up the original dataset :(1) ACTIVITY, (2) TIME, (3) SL, (4) EEG, (5) BP, (6) HR and (7) CIRCULATION. Standing, walking, sitting, falling, cramps, and running are all included in the dataset. Before using the learning method; we set the activity attribute to 1 for falling activities and 0 for all other activities, as shown in table-1.

TABLE I. ATTRIBUTES CONSIDERED FOR SYMPTOMS OF HUMAN FALL

Parameter	Abnormal range	
	Adult male	Adult female
QRS interval	$> 0.12s$	
Heart rate	If not $60 - 100$bpm	
QT interval	$> 0.40s$	
Haematocrit level	If not $42 - 54\%$	If not $38 - 46\%$
Systolic BP (top) Diastolic BP	< 90mmHg	
Anemia	If not $12.4 - 14.9$gm/ If not $11.7 - 13.8$gm/	
Sugar level	If not $95 - 100\%$	
SpO2	dL	dL

B. Gaussian filtering with kernel neural network (GF_KNN) based Pre-processing:

The Gaussian filter is commonly used to blur pictures as well as volumes, while its first and higher order derivatives are frequently employed to detect edges as well as other data features, utilizing ability to suppress high frequency information. Equation (1) defines the 3D Gaussian filter

$$g_{3D}(x,y,z) = \frac{1}{\sqrt{8\pi^3}\sigma_x\sigma_y\sigma_z} e^{-\left(\frac{x^2}{2\sigma_x^2}+\frac{y^2}{2\sigma_y^2}+\frac{z^2}{2\sigma_z^2}\right)}, \quad (1)$$

Where $\sigma$is the Gaussian filter's standard deviation and x, y, and z are voxel coordinates relative to filter's centre. Consider Gaussian is linearly separable; we may do this convolution more quickly, i.e. from equation (2)

$$\begin{aligned} g_{3D}(x,y,z) &= g_{1D}(x)\cdot g_{1D}(y)\cdot g_{1D}(z) \\ &= \frac{1}{\sqrt{2\pi}\sigma_x} e^{-\left(\frac{x^2}{2\sigma_x^2}\right)} \cdot \frac{1}{\sqrt{2\pi}\sigma_y} e^{-\left(\frac{y^2}{2\sigma_y^2}\right)} \cdot \frac{1}{\sqrt{2\pi}\sigma_z} e^{-\left(\frac{z^2}{2\sigma_z^2}\right)} \end{aligned} \quad (2)$$

When it comes to derivatives, any of these terms are substituted with Gaussian's equivalent derivative, as shown in eq. (3):

$$g_{1D}^{x} = \frac{-x}{\sqrt{2\pi}\sigma_x^3} e^{-\left(\frac{x^2}{2\sigma_x^2}\right)} \quad (3)$$

The Gaussian filter has unlimited support in theory. Its magnitude can be ignored at a distance of 3–4 from its centre from a practical standpoint.

$$\sum_{i=0}^{d-1} f_{|i-r|}\cdot v_{p+i-r} = f_0\cdot v_p + \sum_{i=1}^{r} f_i\cdot\left(v_{p+i} \pm v_{p-i}\right) \quad (4)$$

The boundary voxels are handled in various ways, including zero padding, circular border conditions, and mirroring. Mirroring method is always used because it avoids apparent artefacts in the boundary area.

1) Determine the parameter q using σeq(5) as a guide:

$$q = \begin{cases} 0.98711\sigma - 0.96330, & \sigma \geq 2.5 \\ 3.97156 - 4.14554\sqrt{1-0.26891\sigma}, & 0.5 \leq \sigma < 2.5. \end{cases} \quad (5)$$

2) Evaluate filter coefficients $\{b_0, b_1, b_2, b_3, B\}$ from eq. (6):

$$\begin{aligned} b_0 &= 1.57825 + (2.44413q) + (1.4281q^2) + (0.422205q^3), \\ b_1 &= (2.44413q) + (2.85619q^2) + (1.26661q^3), \\ b_2 &= -(1.4281q^2) - (1.26661q^3), \\ b_3 &= 0.422205q^3, \\ B &= 1 - (b_1 + b_2 + b_3)/b_0. \end{aligned} \quad (6)$$

3) Apply a one-dimensional forward difference equation (7):

$$w_n = Ba_n + (b_1 w_{n-1} + b_2 w_{n-2} + b_3 w_{n-3})/b_0, \quad (7)$$

Iterating through input from $n = 0$ up to $n = N - 1$.

4) Apply a backward difference equation:

Using equation (8), create the output c_n from the intermediate result w_n.

$$c_n = Bw_n + (b_1 c_{n+1} + b_2 c_{n+2} + b_3 c_{n+3})/b_0 \quad (8)$$

Iterating through array w_n from $n = N - 1$ down to $n = 0$.

Initiate from a convolution with output f(x), i.e from eq. (9).

$$f(x) = x \oplus w \quad (9)$$

ith element of convolution output f(x) is evaluated as follows: eq. (10):

$$f_i(x) = \langle x_{(i)}, w \rangle \quad (10)$$

We began by defining the index I at zero. The output of kervolution g(x) is defined as eq. (11)

$$g(x) = x \otimes w \quad (11)$$

Where $\otimes$ is kervolutional operator, specifically, eq. (12) is defined as the i^{th} element of g(x):

$$g_i(x) = \langle \varphi(x_{(i)}), \varphi(w) \rangle \quad (12)$$

Definition (4) allows us to extract features in a HD space while simultaneously having a substantially larger computational complexity than the previous definition

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*

(2).Fortunately, we can use the kernel method to avoid having to explicitly calculate HD features ϕ(x(i)), because eq. (13)

$$\langle \varphi(x_{(i)}), \varphi(w) \rangle = \sum_j c_j (x_{(i)}^T w)^j = \kappa(x_{(i)}, w) \quad (13)$$

Mapping function $\phi(\cdot)$ or a predetermined kernel $\kappa(\cdot\ ,\ \cdot)$, such as Gaussian RBF kernel, can calculate coefficient cj, if the feature dimension d is infinite. Kernels with various sizes have numbers of specifications; for example, kernels with sizes of 3×3, 5×5 and 7×7 have 9, 25 and 49 learnable specifications. To summarise, random kernel's advantages are as follows: first, regardless of size of kernel or depth of NNs layers, a random kernel minimizes number of specifications exponentially.

The operation of a kervolutional layer is similar to that of a convolutional layer, but differs slightly from the usual formulation (3), where x(i) forms a 3-D patch in a sliding window on input. We also incorporate all popular accessible convolution structures in CNN library for kervolution, including input and output channels as shown in figure-2, to be compatible with previous studies.

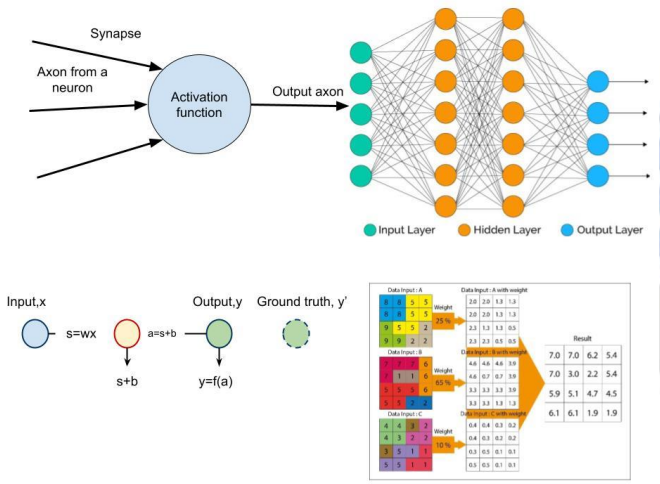


Figure 2. Operation of GF_KNN

We may extract specific types of features via kervolution without worrying about the weight settings. However, as previously said, still essential to modify hyper-parameters for some specific kernels, such as polynomial kernel's balancer cp and Gaussian RBF kernel's smoother, it discovered that systems performance is mainly unaffected by kernel hyper-parameters, it can be inconvenient when we don't know much about the kernel. As a result, we also use back-propagation to train the network with learnable kernel hyper-parameters. Although this theoretically increases training complexity, we discovered that it provides more flexibility in practice, and supplementary cost of training numerous kernel parameters, for example, are obtained as eq. (14)

$$\frac{\partial}{\partial w}\kappa_g(x,w) = 2\gamma_g(x-w)\kappa_g(x,w)\frac{\partial}{\partial \gamma_g}\kappa_g(x,w) = -\| x - w \|^2\ \kappa_g(x,w) \quad (14)$$

C. Encoded Stacked Deconvolutional Network (EnSt-DeConvNet):

To collect contextual data as well as refine weak object delineation, we create a shallow de-convolutional network (SDN) unit, whose topology is shown in Fig. 3. (a).An encoder module and a decoder module are included in an SDN unit. Stack 2 down sampling blocks in an encoder module to extend the network's receptive fields. Up sampling blocks are employed twice in a decoder module to attain a more advanced reconstruction of feature maps.

Utilize down sampling blocks twice in this case, resulting in a spatial resolution of 1/ 16 for the input image. Figure 3(b) depicts the structure of the down sampling block. A max-pooling layer, two convolution layers, and a compression layer make up a down sampling block.

Second, we cascade two convolution layers with intra-unit connections behind max-pooling layer, which is helpful to feature reuse. Intra-unit connections result in a linear increase in channel number of feature maps, putting a strain on GPU memory.

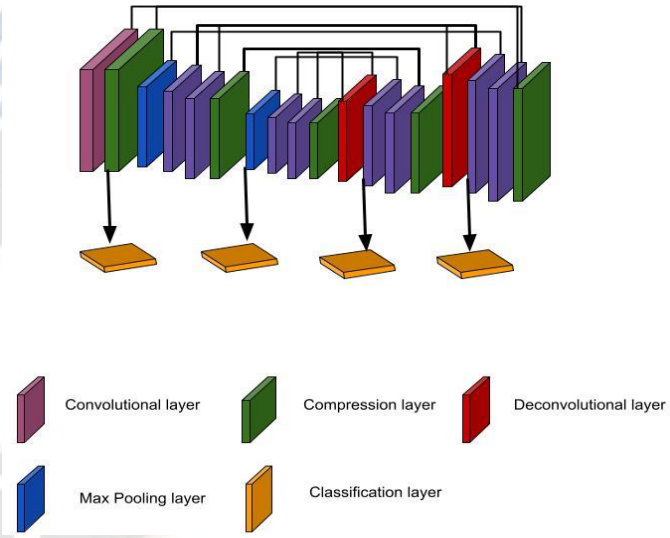


Figure 3. Structure of proposed SDN, lower part indicates detailed structure of SDN unit (a), downsampling block (b), upsampling block (c).

Finally, to lower computational cost as well as memory demand, use a compression layer to build a feature map F_n^i by performing convolutions with fewer filters to minimize channel number of feature map Q_n^i.In a nutshell, the i'^{th}down sampling block takes two feature maps as inputs, $F_n^{i-1}$and F_{n-1}^i and generates a new feature map F_n^i. Operations are summed up as follows: eq. (15):

$$P_n^i = \text{Max}\,(F_n^{i-1})$$

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*

$$Q_n^i = \text{Trans}\left([O_n^i, H_1^k]\right) \qquad (15)$$

$$F_n^i = \text{Comp}\,(Q_n^i)$$

Where $P_n^i, (F_n^{i-1})$ signifies a max-pooling operation and $P_n^i = \text{Max}\ (F_n^{i-1})$ denotes concatenation of feature maps. Comp() is a three-dimensional convolution operation. Feature map O_n^i is then concatenated with feature map H_1^k from kth block in first SDN unit's encoder module, where k th block has same resolution as O_n^i.Equation (16) can be used to calculate the output F_n^i of the I th up sampling block:

$$O_n^i = \text{Deconv}\,(F_n^{i-1})Q_n^i = \text{Trans}\left([O_n^i, H_1^k]\right) \qquad (16)$$

$$F_n^i = \text{Comp}\,(Q_n^i)$$

Note that our SDN unit's greatest resolution is set to a fifth of the input photos. One major reason for this architecture is that it allows us to stack more SDN units while using less GPU RAM.

$$h^k = \sigma\left(x * W^k + b^k\right) \qquad (17)$$

Where σ is an activation function and $*$ indicates 2D convolution, and is a bias that is propagated to entire map. Because each filters to focus on features of entire input, we apply a single bias per latent map. Rebuilding is done with the help of eq. (18)

$$y = \sigma\left(\sum_{k\in H} h^k * \grave{W}^k + c\right) \qquad (18)$$

There is one bias c per input channel once more. W˜ recognizes flip operation across both dimensions of weights; H identifies set of latent feature mappings.

IV. PERFORMANCE ANALYSIS:

The Experimental Setup is described below. We utilized Python 3 as our programming language and Anaconda's Spyder (version 3.3.4) as our working IDE. For computational tasks and working with files like comma-separated value (CSV) files, Numpy and Pandas libraries were employed. MatPlot package is used to display the findings and plotted graphs. The Time module was utilized to keep track of start and end times as well as determine the execution time for techniques.

A. Dataset description

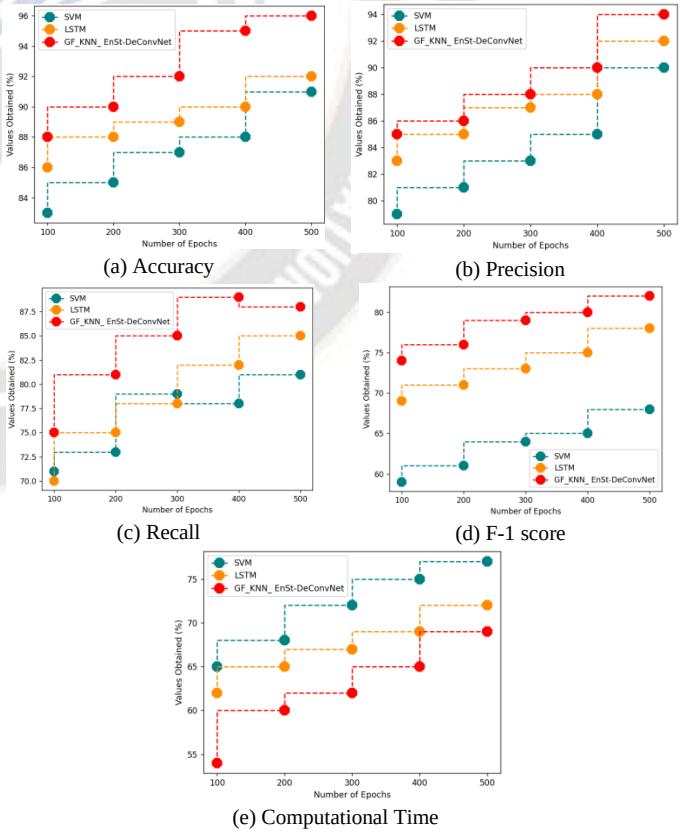
We utilized a dataset from Kaggle, a Google-owned online community for data scientists that allows users to download as well as upload datasets. Kaggle was created in April 2010 and allows users to download and upload datasets. Dataset in question was produced by gathering physiological data from elderly individuals at several Chinese hospitals. This dataset, however, cannot be used directly.

This study used data from a publicly available dataset. Both the PhysioNet (DOI: 10.13026/ C2WW2W) and Figshare (DOI: 10.6084/ m9. figshare. 3394432) websites have access to it. The study that produced the dataset, as well as a thorough description of it, may be found here (Santos & Duarte, 2016).Information was gathered from 163 male and female volunteers of various ages and health statuses. Each participant performed a single 1-2-hour study session during which all data was collected.

TABLE II. COMPARATIVE ANALYSIS OF DATASETS BETWEEN EXISTING AND PROPOSED TECHNIQUE

Data sets	Techniques	Accuracy	Precision	Recall	Computational time	F1-score
Kaggle	SVM	91	90	81	77	68
	LSTM	92	92	85	72	78
	GF_KNN_EnSt-DeConvNet	96	94	88	69	82
FigShare	SVM	89	88	85	80	78
	LSTM	90	91	89	72	82
	GF_KNN_EnSt-DeConvNet	92	92	91	69	84

The table-1 shows comparative analysis for parameters in terms of accuracy, precision, recall, F-1 score and computational time between proposed and existing techniques for Kaggle and Figshare dataset.Here the existing techniques compared are SVM, LSTM.



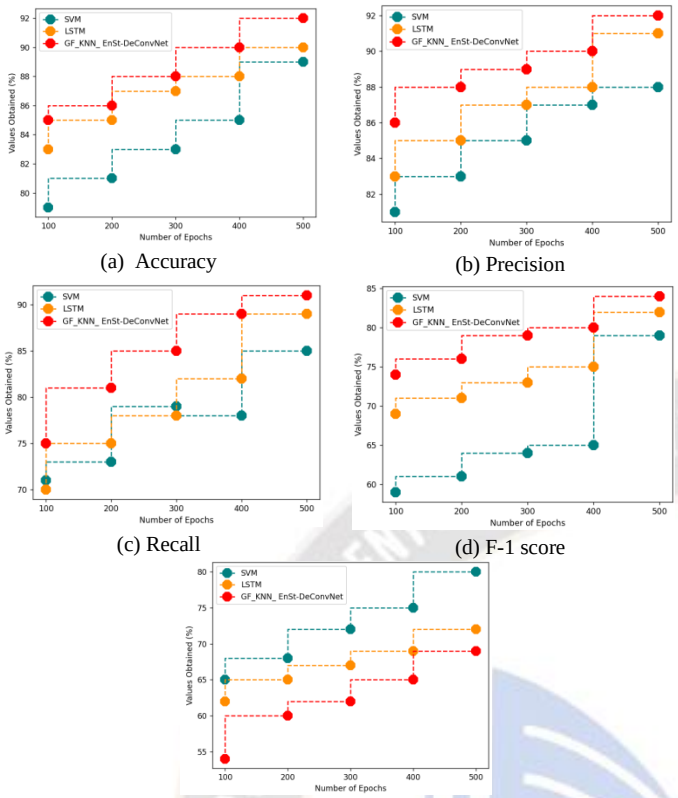
(a) Accuracy (b) Precision

(c) Recall (d) F-1 score

(e) Computational Time

Figure 4. Comparative analysis of Kaggle dataset in terms of (a) accuracy, (b) precision, (c) recall, (d) F-1 score (e) computational time

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*



(a) Accuracy (b) Precision

(c) Recall (d) F-1 score

(e) Computational Time

Figure 5. Comparative analysis of Figshare dataset in terms of (a) accuracy, (b) precision, (c) recall, (d) F-1 score (e) computational time

Above figure 4, 5 shows parametric comparison for proposed and existing techniques for Kaggle and Figshare based input dataset. For the Kaggle dataset, the proposed technique obtained accuracy of 96%, Precision of 94%, Recall of 88% and F-1 score of 82%, computational time of 69%. These parametric results are optimal when compared with this existing technique for Kaggle dataset. At the same time, the proposed technique obtained accuracy of 92%, Precision of 92%, Recall of 91% and F-1 score of 84%, computational time of 69% for Figshare dataset. Here, the proposed technique obtained optimal results compared with the existing technique for both the datasets.

V. CONCLUSION:

This research proposed analysis in symptoms for human fall based on deep learning techniques in processing sensor data. Here the input data has been collected as the patient's pre-historic symptoms for fall and monitored sensor data. This data has been pre-processed and segmented based on neural network techniques in which the data has been normalized and trained in analyzing the symptoms for fall. The pre-processing has been carried out using Gaussian filtering with kernel neural network (GF_KNN) and this normalized trained data has been segmented using encoded Stacked De-convolutional Network (EnSt-DeConvNet). The experimental results shows parametric analysis of proposed fall symptom analysis in terms of accuracy of 96%, Precision of 94%, Recall of 88% and F-1 score of 82%, computational time of 69%.

REFERENCES

[1] Santos, G. L., Endo, P. T., Monteiro, K. H. D. C., Rocha, E. D. S., Silva, I., & Lynn, T. (2019). Accelerometer-based human fall detection using convolutional neural networks. Sensors, 19(7), 1644.

[2] Shu, F., & Shu, J. (2021). An eight-camera fall detection system using human fall pattern recognition via machine learning by a low-cost android box. Scientific reports, 11(1), 1-17.

[3] Kong, Y., Huang, J., Huang, S., Wei, Z., & Wang, S. (2019). Learning spatiotemporal representations for human fall detection in surveillance video. Journal of Visual Communication and Image Representation, 59, 215-230.

[4] Savadkoohi, M., Oladunni, T., & Thompson, L. A. (2021). Deep neural networks for human's fall-risk prediction using force-plate time series signal. Expert Systems with Applications, 182, 115220.

[5] Saxena, U., Moulik, S., Nayak, S. R., Hanne, T., & Sinha Roy, D. (2021). Ensemble-Based Machine Learning for Predicting Sudden Human Fall Using Health Data. Mathematical Problems in Engineering, 2021.

[6] Souza, P. V. C., Guimaraes, A. J., Araujo, V. S., Batista, L. O., & Rezende, T. S. (2020). An interpretable machine learning model for human fall detection systems using hybrid intelligent models. In Challenges and Trends in Multimodal Fall Detection for Healthcare (pp. 181-205). Springer, Cham.

[7] Ali, S. F., Khan, R., Mahmood, A., Hassan, M. T., & Jeon, M. (2018). Using temporal covariance of motion and geometric features via boosting for human fall detection. Sensors, 18(6), 1918.

[8] Mauldin, T. R., Canby, M. E., Metsis, V., Ngu, A. H., & Rivera, C. C. (2018). SmartFall: A smartwatch-based fall detection system using deep learning. Sensors, 18(10), 3363.

[9] Soni, M., Chauhan, S., Bajpai, B., & Puri, T. (2020, September). An approach to enhance fall detection using machine learning classifier. In 2020 12th International Conference on Computational Intelligence and Communication Networks (CICN) (pp. 229-233). IEEE.

[10] Zhang, J., Wu, C., & Wang, Y. (2020). Human fall detection based on body posture spatio-temporal evolution. Sensors, 20(3), 946.

[11] Xu, Q., Huang, G., Yu, M., & Guo, Y. (2020). Fall prediction based on key points of human bones. Physica A: Statistical Mechanics and its Applications, 540, 123205.

[12] Singh, K., Rajput, A., & Sharma, S. (2019). Human fall detection using machine learning methods: a survey. Int. J. Math. Eng. Manag. Sci, 5, 161-180.

[13] Nizam, Y., Mohd, M. N. H., & Jamil, M. (2018). Development of a user-adaptable human fall detection based on fall risk levels using depth sensor. Sensors, 18(7), 2260.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 10 Issue: 12
DOI: https://doi.org/10.17762/ijritcc.v10i12.5915
***Article Received**: 16 October 2022 **Revised**: 20 November 2022 **Accepted**: 18 December 2022*

[14] Sarabia-Jácome, D., Usach, R., Palau, C. E., &Esteve, M. (2020). Highly-efficient fog-based deep learning AAL fall detection system. Internet of Things, 11, 100185.

[15] Espinosa, R., Ponce, H., Gutiérrez, S., Martínez-Villaseñor, L., Brieva, J., & Moya-Albor, E. (2019). A vision-based approach for fall detection using multiple cameras and convolutional neural networks: A case study using the UP-Fall detection dataset. Computers in biology and medicine, 115, 103520.

[16] Sampath Dakshina Murthy, A., Karthikeyan, T., & Vinoth Kanna, R. (2021). Gait-based person fall prediction using deep learning approach. Soft Computing, 1-9.

[17] Panahi, L., &Ghods, V. (2018). Human fall detection using machine vision techniques on RGB–D images. Biomedical Signal Processing and Control, 44, 146-153.

[18] Fan, K., Wang, P., & Zhuang, S. (2019). Human fall detection using slow feature analysis. Multimedia Tools and Applications, 78(7), 9101-9128.

[19] Delgado-Escano, R., Castro, F. M., Cozar, J. R., Marin-Jimenez, M. J., Guil, N., &Casilari, E. (2020). A cross-dataset deep learning-based classifier for people fall detection and identification. Computer methods and programs in biomedicine, 184, 105265.

[20] Ribeiro, O., Gomes, L., & Vale, Z. (2022). IoT-Based Human Fall Detection System. Electronics, 11(4), 592.

[21] Hadjadji, B., Saumard, M., & Aron, M. (2022). Multi-oriented run length based static and dynamic features fused with Choquet fuzzy integral for human fall detection in videos. Journal of Visual Communication and Image Representation, 82, 103375.

[22] Paul Ijjina, E. (2022). Human Fall Detection in Depth-Videos Using Temporal Templates and Convolutional Neural Networks. In Advances in Assistive Technologies (pp. 217-236). Springer, Cham.

[23] Yadav, S. K., Luthra, A., Tiwari, K., Pandey, H. M., & Akbar, S. A. (2022). ARFDNet: An efficient activity recognition & fall detection system using latent feature pooling. Knowledge-Based Systems, 239, 107948.

[24] Ezatzadeh, S., Keyvanpour, M. R., &Shojaedini, S. V. (2021). A human fall detection framework based on multi-camera fusion. Journal of Experimental & Theoretical Artificial Intelligence, 1-20.