**Progress Report Draft | 04 April 2024**

1. **Objective**

This report is a replication and experimentation of the research paper, “A vision-based approach for fall detection using multiple cameras and convolutional neural networks: A case study using the UP-Fall detection dataset”, authored by R Espinosa et al., which used optical flow and windowing technique for fall detection.

This project will focus on building upon work done reproducing the original model’s methodology and architecture, assessing its performance, and interpreting the potential applications in our project. This includes exploring alternative methods of data preprocessing for optical flow and experimenting with 3D Convolutional Neural Network architectures to potentially enhance model accuracy and performance.

This project aims to perform a comparative analysis of various preprocessing techniques and neural network architecture designs to understand the feasibility of an improvement in the performance of the fall detection model in a vision-based approach using the UP-Fall detection dataset (Martínez-Villaseñor et al., 2019). The main objective is to identify the impact of these preprocessing techniques and architecture designs in improving the model performance while simultaneously reducing the time complexity.

1. **Neural Network Architecture**
   1. **3D CNN Model (ReLU)**

The architecture of the 3D CNN model follows the same architecture as the 2D CNN model with 2D layers being changed to 3D layers.

* **3D Convolutional Layers for Feature Extraction:**

Layer 1 consisting of 128 convolution filters with a kernel size of 3 x 3 x 3.

Layer 2 consisting of 128 convolution filters with a kernel size of 3 x 3 x 3.

Layer 3 consisting of 64 convolution filters with a kernel size of 3 x 3 x 3.

* **Max-Pooling Layers:**

3D max-pooling layers are used after each convolutional layer to reduce spatial dimensions.

* **Fully Connected Layers for Fall Detection**

Layer 1 consisting of 64 ReLU units.

Layer 2 consisting of 128 ReLU units.

Layer 3 consisting of 254 ReLU units.

* **Output Layer:**

2D SoftMax layer with a single binary output:

* Fall (represented as 1)
* No fall (represented as 0)
  1. **3D CNN Model (Gaussian Error Linear Unit - GeLU)**

The architecture of this model follows the same architecture as the previous 3D CNN model with the ReLU units swapped for GeLU.

* **3D Convolutional Layers for Feature Extraction:**

Layer 1 consisting of 128 convolution filters with a kernel size of 3 x 3 x 3.

Layer 2 consisting of 128 convolution filters with a kernel size of 3 x 3 x 3.

Layer 3 consisting of 64 convolution filters with a kernel size of 3 x 3 x 3.

* **Max-Pooling Layers:**

3D max-pooling layers are used after each convolutional layer to reduce spatial dimensions.

* **Fully Connected Layers for Fall Detection**

Layer 1 consisting of 64 GeLU units.

Layer 2 consisting of 128 GeLU units.

Layer 3 consisting of 254 GeLU units.

* **Output Layer:**

2D SoftMax layer with a single binary output:

* Fall (represented as 1)
* No fall (represented as 0)
  1. **3D Sequential CNN and LSTM (Long Short-Term Memory) Model**

The architecture of this model consists of the Optical Flow data being sequentially passed through a CNN and then through a LSTM.

* **3D Convolutional Layers for Feature Extraction:**

Layer 1 consisting of 64 convolution filters with a kernel size of 3 x 3 x 3. Followed by a Batch Normalization layer.

Layer 2 consisting of 128 convolution filters with a kernel size of 3 x 3 x 3. Followed by a Batch Normalization layer.

Layer 3 consisting of 256 convolution filters with a kernel size of 3 x 3 x 3. Followed by a Batch Normalization layer.

Layer 3 consisting of 256 convolution filters with a kernel size of 3 x 3 x 3. Followed by a Batch Normalization layer.

* **Max-Pooling Layers:**

3D max-pooling layers are used after each convolutional layer to reduce spatial dimensions.

* **Global Average Pooling Layer:**

A global average pooling layer is utilized after the last convolutional layer to reduce the dimensions and extract the most relevant features.

* **Long Short-Term Memory**

LSTM Layer (256)

Dropout

Fully Connected Layer (2)

* **Output Layer:**

A linear layer with 2 units giving a single binary output:

* Fall (represented as 1)
* No fall (represented as 0)

1. **Results**
   1. **Metrics**

The metrics used to evaluate the model were Accuracy, Precision, Recall, Specificity, and F1-Score as shown in the descriptions below.

* **Accuracy**

The ratio of correctly predicted instances to the total instances

* **Precision**

The ratio of correctly predicted positives observations to the total predicted positives

* **Recall (Sensitivity)**

The ratio of correctly predicted positives observations to all the actual positives

* **Specificity**

The ratio of correctly predicted negative observations to all the actual negatives.

* **F1-Score**

The weighted average of Precision and Recall

1. **Results and Evaluation** 
   1. **Comparative Analysis**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data** | **Accuracy** | **Precision** | **Recall** | **Specificity** | **F1-Score** | **Loss** |
| **Research** | **95.64** | **96.91** | **97.95** | **83.08** | **97.43** |  |
| **Replication** | **98.98** | **84.14** | **81.94** | **99.51** | **83.02** |  |
| **3D CNN BGS Canny**  LR [0.0001] E [50] | 94.50 | 35.64 | 98.85 | 94.37 | 52.39 | 57.72 |
| **BGS findcon()**  LR [0.0001] E [50] | 93.63 | 32.33 | 99.01 | 93.46 | 48.75 | 73.02 |
| **BGS findcon()**  LR [0.0001] E [100] | 93.96 | 33.46 | 98.59 | 93.81 | 49.96 | 35.76 |
| **BGS findcon()**  LR [0.00001] E [50] | 94.36 | 34.97 | 98.28 | 94.23 | 51.59 | 35.04 |
| **BGS findcon()**  LR [0.0001] E [50] **GeLU** | 93.62 | 32.28 | 98.85 | 93.45 | 48.66 | 69.69 |
| **Sequential CNN with LSTM LR [0.0001]**  **E [50]** | 96.09 | 13.66 | 95.31 | 96.09 | 23.89 | 16.85 |

Loss: 0.7302, Accuracy: 0.9363, Precision: 0.3233, Recall: 0.9901, Specificity: 0.9346, F1-Score: 0.4875 // Time: 745.56827 sec. [12.42 min]

Loss: 0.3576, Accuracy: 0.9396, Precision: 0.3346, Recall: 0.9859, Specificity: 0.9381, F1-Score: 0.4996 // Time: 1333.77224 sec. [ 22.22 min]

Loss: 0.3504, Accuracy: 0.9436, Precision: 0.3497, Recall: 0.9828, Specificity: 0.9423, F1-Score: 0.5159 // Time taken: 741.416862 sec. [12.35 min]

Loss: 0.6969, Accuracy: 0.9362, Precision: 0.3228, Recall: 0.9885, Specificity: 0.9345, F1-Score: 0.4866 // Time taken: 743.8618334999992 sec. [12.39 min]

Loss: 0.1692, Accuracy: 0.9576, Precision: 0.4186, Recall: 0.9922, Specificity: 0.9565, F1-Score: 0.5888 // Time taken: 578.91029 sec. [9.6485 min]

* 1. **Graphs**
     1. **BGS findcontours()** LR[0.0001] E[50]

A blue and white graph

Description automatically generated

Figure 1 Confusion Matrix - BGS findcontours() LR[0.0001] E[50]

A group of graphs on a white background

Description automatically generated

Figure 2 Results - BGS findcontours() LR[0.0001] E[50]

* + 1. **BGS findcontours()** LR[0.0001] E[100]

A blue and white graph

Description automatically generated

Figure 3 Confusion Matrix - BGS findcontours() LR[0.0001] E[100]

A graph of a graph

Description automatically generated with medium confidence

Figure 4 Results - BGS findcontours() LR[0.0001] E[100]

* + 1. **BGS findcontours()** LR[0.00001] E[50]

A blue and white graph

Description automatically generated

Figure 5 Confusion Matrix - BGS findcontours() LR[0.00001] E[50]

A graph of a graph

Description automatically generated with medium confidence

Figure 6 Results - BGS findcontours() LR[0.00001] E[50]

* + 1. **BGS findcontours() + GeLU** LR[0.0001] E[50]

A blue and white graph

Description automatically generated

Figure 7 Confusion Matrix - BGS findcontours() LR[0.0001] E[50] GeLU

A graph of a graph

Description automatically generated with medium confidence

Figure 8 Results - BGS findcontours() LR[0.0001] E[50] GeLU

* + 1. **Sequential CNN + LSTM** LR[0.0001] E [50]

A blue and white graph

Description automatically generated

Figure 9 Confusion Matrix - Sequential CNN + LSTM LR[0.0001] E[50]

A graph of a graph

Description automatically generated with medium confidence

Figure 10 Results - Sequential CNN + LSTM LR[0.0001] E[50]