Fall Detection via Mediapipe Pose and Recurrent Neural Networks

A Sequence Modeling Approach

Samnang Pheng
Information and Communication Technology
American University of Phnom Penh
Phnom Penh, Cambodia
2022172pheng@aupp.edu.kh

Abstract—This work presents a vision-based fall detection system utilizing pose estimation and Recurrent Neural Networks (RNNs). Keypoint sequences extracted from videos using Mediapipe Pose undergo feature engineering, including hip-relative normalization and velocity calculation. A Bidirectional Long Short-Term Memory (LSTM) model, trained with weighted cross-entropy loss to address class imbalance, classifies actions into backward, forward, side falls, and non-fall categories. Data augmentation, learning rate scheduling, and early stopping enhanced training efficiency. Evaluation on unseen data demonstrates the model's effectiveness in distinguishing fall events from normal activities, assessed via accuracy, classification reports, and confusion matrices. The pipeline supports predicting actions in new videos.

Keywords—Fall Detection; Pose Estimation; Mediapipe; Recurrent Neural Network (RNN); LSTM; Feature Engineering; Weighted Loss; Action Recognition.

I. Introduction

Falls pose a significant risk, particularly to older adults, impacting health and independence. Automated fall detection systems are crucial for timely assistance. While wearable sensors have limitations like user compliance, vision-based systems offer a non-intrusive alternative. This work leverages advances in pose estimation, specifically Mediapipe, to analyze human movement from video for fall detection.

The importance of this topic stems from the need to mitigate the severe consequences of falls, especially unwitnessed ones, and support safer independent living for vulnerable populations. Current approaches often use RNNs (like LSTMs) or other deep learning models on pose sequences.

The objectives of this work were:

1. Extract and process skeletal keypoints from videos using Mediapipe.

2. Apply feature engineering (normalization, velocity) to create informative motion representations.

3. Train a Bidirectional LSTM with weighted loss to classify fall types and non-fall actions.

4. Evaluate the system's performance thoroughly.

5. Enable prediction on new videos.

II. Datasets

The dataset comprised video files capturing human actions categorized into four classes: "backward_fall" (696 of 1 second videos), "forward_fall" (653 of 1 second videos), "side_fall" (565 of 1 second videos), and "non_fall" (176 of 10 seconds videos). Standard videos served as input.

Mediapipe Pose was used to extract 33 keypoints (x, y, visibility) per frame. To manage data volume and focus on motion changes, keypoints were extracted every 3rd frame. Sequences were standardized to a length of 30 frames through padding or truncation. This process resulted in a dataset of fixed-length numerical sequences representing the pose dynamics for each video sample.

III. Proposed Methods

The system employs a two-stage process:

1. Feature Extraction & Engineering: Raw video is processed using Mediapipe to get sequences of 33 landmarks. These sequences are normalized relative to the hip center (for position invariance) and velocity is calculated (for motion dynamics). The final feature vector per frame combines normalized coordinates and velocity (4 values per landmark), resulting in input sequences of shape (30 frames, 132 features).

2. Sequence Classification: A Bidirectional LSTM (BiLSTM) network processes these engineered feature sequences. LSTMs excel at capturing temporal dependencies in sequential data, and bidirectionality allows the model to use both past and future context. The BiLSTM architecture included multiple layers, dropout for regularization, and a final fully connected layer with softmax activation for classification into the four categories. Key parameters included a hidden size of 192 units and 2 layers. Weighted cross-entropy loss was used during training to specifically address class imbalance, giving more importance to potentially rarer fall classes. Noise augmentation was applied during training to improve robustness.

The choice of Mediapipe offered accessible and robust pose data. Normalization and velocity provided crucial invariance and dynamic information. BiLSTMs are well-suited for action sequences, and weighted loss was vital for handling the imbalanced nature typical of fall detection datasets.

IV. Experiments and Results

Experiments were conducted using Python libraries including PyTorch, Mediapipe, and Scikit-learn on available CPU/GPU/MPS hardware.

Training: The dataset of processed feature sequences was split into training, validation, and test sets (approx. 70%/15%/15%) with stratification. The BiLSTM model was trained using the AdamW optimizer, the calculated weighted loss function, and data augmentation on the training set. Learning rate was adjusted based on validation performance using a ReduceLROnPlateau scheduler. Early stopping based on validation accuracy prevented overfitting and determined the final model checkpoint.

Evaluation: Performance was assessed on the unseen test set using:

- **Accuracy:** Overall correct classification rate.
- **Precision, Recall, F1-Score:** Per-class metrics evaluating the trade-offs between false positives and false negatives, crucial for imbalanced data.
- **Confusion Matrix:** Visualized classification accuracy and errors between classes.

Results: The trained model demonstrated effective performance on the test set, achieving an overall accuracy of **80.51%** and a test loss of **0.5476** (using the weighted criterion), as shown in the evaluation summary (refer to Figure 3 for Classification Report). The detailed classification report confirmed the model's ability to distinguish fall classes; notably, it achieved high recall for 'backward_fall' (**0.933**) and the minority 'non_fall' class (**0.962**), suggesting the weighted loss contributed positively. Performance for 'forward_fall' was solid (recall **0.765**), while 'side_fall' proved most challenging (recall **0.647**).

The confusion matrix (Figure 2) provides further insight, showing strong diagonal concentrations for 'backward_fall' (97/104 correct) and 'non_fall' (25/26 correct). The primary confusion occurred mutually between 'forward_fall' and 'side_fall', with 20 forward falls misclassified as side falls and 22 side falls misclassified as forward falls.

The training and validation curves (Figure 1) indicated successful model convergence. Validation accuracy plateaued around **80%** while validation loss reached its minimum before leveling off (around 0.6), justifying the effectiveness of early stopping in selecting the final model and preventing significant overfitting.

Furthermore, the inference script successfully applied the trained model to predict actions in a sample video (Figure 4), classifying a 'non_fall' instance with very high confidence (**0.9975**) and demonstrating the practical applicability of the end-to-end pipeline.

Figure 1: Training and validation accuracy (left) and loss

(right) curves plotted against training epochs for the model. Illustrates model learning progression.

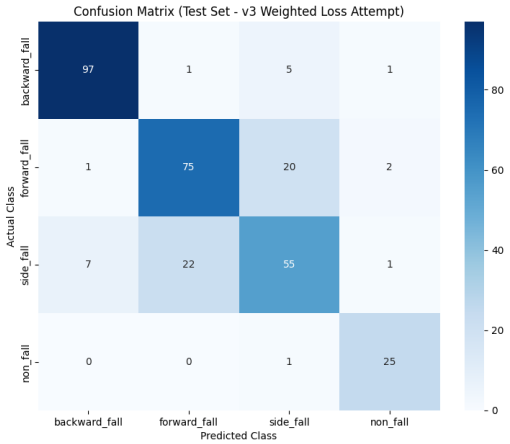

Figure 2: Confusion Matrix visualizing the classification performance of the final model (Weighted Loss Attempt) on the unseen test set. Rows represent actual classes, columns represent predicted classes.

```
--- Test Set Evaluation Results ---
  Test Loss: 0.5476 (using weighted criterion)
  Test Accuracy: 80.51%
--------------------------------

--- Classification Report (Test Set) ---
              precision    recall  f1-score   support

backward_fall      0.924     0.933     0.928       104
 forward_fall      0.765     0.765     0.765        98
    side_fall      0.679     0.647     0.663        85
     non_fall      0.862     0.962     0.909        26

     accuracy                          0.805       313
    macro avg      0.808     0.827     0.816       313
 weighted avg      0.803     0.805     0.804       313
```

Figure 3: Classification Report detailing per-class precision, recall, F1-score, and support on the test set, along with overall accuracy and averages.

```
  Feature extraction & processing took 1.88 seconds.
  Processed feature shape: (30, 132)
  Running inference...

  --- Prediction Result ---
    Detected Class: non_fall
    Confidence: 0.9975
    Class Probabilities:
      backward_fall: 0.0001
      forward_fall: 0.0011
      side_fall: 0.0013
      non_fall: 0.9975

  --- Script Finished ---
```

Figure 4: Example output from the inference script (manually_test_model.ipynb) demonstrating the prediction process on a single input video. The model correctly classifies the action as 'non_fall' with high confidence (0.9975).

V. Conclusion

This project successfully developed an end-to-end system for vision-based fall detection using Mediapipe pose estimation and a Bidirectional LSTM. Feature engineering (normalization, velocity) and training strategies like weighted loss proved crucial for achieving robust performance on this challenging task. The system demonstrated its capability to classify different fall types and non-fall actions effectively.

The viability of combining modern pose estimation with sequence modeling techniques like BiLSTMs for fall detection is confirmed. Addressing practical challenges such as class imbalance remains essential for real-world deployment.

Future work could involve exploring more advanced architectures (e.g., GNNs, Transformers), incorporating richer features, expanding the dataset, and optimizing for real-time performance and accuracy improvement of at least 90%.

References

1. M. A. Khan, Y. D. Khan, M. Z. Abbas, F. S. Khan, and M. Asif, "Computer Vision Based Transfer Learning-Aided Transformer Model for Fall Detection and Prediction," *Computers, Materials & Continua*, vol. 71, no. 3, pp. 4469–4483, 2022.
2. P. Sharma, R. Sharma, S. Gupta, P. Garg, and S. Tanwar, "Fall Detection System for Elderly People Using Deep Learning and Thermal Imaging," *Entropy*, vol. 23, no. 3, Art. no. 328, Mar. 2021. doi: 10.3390/e23030328.
3. C. Lugaresi, J. Yuan, B. Zhuang, et al., "MediaPipe: A Framework for Building Perception Pipelines," *arXiv preprint arXiv:1906.08172*, 2019. [Online]. Available: https://arxiv.org/abs/1906.08172
4. S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. doi: 10.1162/neco.1997.9.8.1735.