Implementation and Reproducibility Notes on EMPATH: Enhancing Word-Level Sign Language Recognition

Kazi Reyazul Hasan[0009-0007-9156-1565] and Muhammad Abdullah Adnan[0000-0003-3219-9053]

Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh
{kazireyazulhasan,abdullah.adnan}@gmail.com

Abstract. This companion paper provides a detailed account of the EMPATH framework[6], which integrates Ensemble Learning, MediaPipe Holistic for gesture tracking, and an Attention-based Transformer model, along with practical insights on designing an effective recognition model. It focuses on the architecture, workflow and offers an in-depth explanation of the interpolation method for handling missing hand keypoints. Additionally, the paper highlights key limitations encountered during development and implementation, providing valuable insights for enhancing the reproducibility and performance of word-level sign language recognition using EMPATH.

1 Codebase

The source code is accessible at https://github.com/kreyazulh/EMPATH. he repository includes not only the codebase but also links to pretrained models and preprocessed files used in many of our experiments. Whether users want to replicate our experiments or start from scratch, all essential details are provided in the repository's README.md file. Additionally, the code is thoroughly commented to enhance readability and understanding.

The "Model for Missing Keypoints" folder contains the interpolation model codes layered on top of MediaPipe. Inside each named folder, there are notebooks for both training and testing. Since preprocessing steps may vary depending on the dataset structure, users should follow the format in the generated CSV files to align with EMPATH's input requirements. An example preprocessing workflow is provided in the "preprocess" folder.

2 Architecture of the Transformer Model in EMPATH

The EMPATH framework employs a sophisticated Transformer architecture to enhance the recognition of Bangla Sign Language (BdSL) and to generalize well in other sign languages. This architecture efficiently processes spatial-temporal data from gestures and movements using attention mechanisms and layered structures.

2.1 Key Components of the Transformer Model

- **Multi-Head Attention Mechanism:** The core of the Transformer, allowing the model to focus on different parts of the input simultaneously.
  - *Changing the number of attention heads alters the model's ability to capture diverse features.*
- **Layer Normalization:** Stabilizes and accelerates training by normalizing outputs.
  - *Changing the normalization technique may affect convergence rates.*
- **Feed-Forward Network (FFN):** Consists of two Dense layers with GELU activation.
  - *Increasing the width of the FFN layers enhances capacity but increases computational cost.*
- **Residual Connections:** Facilitates gradient flow by adding the input to the output of each layer.
  - *Removing residual connections could hinder learning by making the model harder to optimize.*
- **Positional Encoding:** Added to embeddings to retain sequence order information.
  - *Changing the positional encoding method impacts how the model understands spatial relationships.*
- **Embedding Layers:** Separate layers for different types of landmarks, handling missing data with special embeddings.
  - *Altering the handling of missing data may lead to inaccuracies in representation.*
- **Output Layer:** Aggregates results from multiple Transformer blocks for predictions.
  - *Changing the final output layer type affects the nature of the predictions (e.g., regression vs. classification).*

2.2 Overall Architecture Flow

The data flow through the EMPATH Transformer model is as follows:

1. **Input Preparation:** Pre-processed landmark data is passed into the Landmark Embedding layers.
2. **Embedding and Positional Encoding:** Separate embeddings are created for each landmark type with positional information added.
3. **Transformer Blocks:** Embedded inputs are processed through multiple Transformer blocks, refining representations iteratively.
4. **Final Output:** Output embeddings from the last block are aggregated to produce predictions for sign language recognition.

This architecture captures details of gestures while maintaining a balance between efficiency and accuracy, achieving state-of-the-art performance in sign language recognition.

3 EMPATH Workflow

The EMPATH framework is designed to enhance the recognition of sign language by processing videos to extract body keypoints and training a transformer-based model to classify gestures. Below is the workflow detailing the steps involved from data preprocessing to final inference.

Algorithm 1 EMPATH Workflow

```
 1: Input: Raw videos of sign language gestures
 2: Output: Predicted labels for the gestures

 3: Preprocessing Phase
 4:     1.1 Load the raw video dataset.
 5:     1.2 Split the dataset into train and test sets.
 6:     1.3 Generate CSV file containing the train-test split and corresponding labels.

 7: Keypoint Extraction Phase
 8:     2.1 For each subset, load the corresponding videos.
 9:     2.2 Use MediaPipe Holistic to extract pose, hand, and face keypoints from each
        video.
10:     2.3 Apply interpolation techniques to handle missing or incomplete hand key-
        points.
11:     2.4 Store the extracted keypoints in Parquet files for both train and test sets.

12: Training Phase
13:     3.1 Load the Parquet files containing keypoints for the train set.
14:     3.2 Train the transformer model using the extracted keypoints.
15:     3.4 Perform ensembling to improve model generalization.
16:     3.5 Save the trained model for inference.

17: Inference Phase
18:     4.1 Load the Parquet files for the test set.
19:     4.2 Use the trained transformer model to predict gestures from the test data.
```

4 Advantages of Pose Extraction

Pose extraction models such as **MediaPipe**[8], **OpenPose**[3], **YOLOv8**[9] and **BlazePose**[1] extract only the relevant body structure keypoints (pose, hand, and face) necessary for gesture recognition. By focusing on specific body structures, these methods eliminate background noise and irrelevant image details typically encountered in image embedding techniques, thereby improving both the accuracy and efficiency of the recognition process.

5 Interpolation Model Analysis

In our work, we developed an interpolation model designed to generate missing hand keypoints that may be absent due to abrupt movements during sign language gestures. This model aims to enhance the continuity and accuracy of gesture recognition, particularly for longer sequences where keypoint loss can significantly impact performance.

To evaluate the effectiveness of our interpolation model, we transitioned from word-level assessments to sentence-level datasets. We specifically compared our results against established datasets such as the RWTH-PHOENIX-Weather-2014T [2] and the How2Sign [5] datasets.

5.1 Improving Keypoints

Employing the default MediaPipe Holistic model on the mentioned dataset samples, we present a more detailed breakdown of the findings in Table 1. Specifically, PHOENIX14T exhibits 30.21% of frames with incomplete keypoints, How2Sign shows 8.63% of frames with missing keypoints. This substantial loss of data poses a challenge when training any skeleton-based model. Moreover, it introduces inconsistency in the testing data, affecting the overall robustness and reliability of the model.

Table 1: Frames with Missing Hand keypoints Using MediaPipe in Different Dataset Sample Clips

Source	Length	Resolution	Total Frames	Missed Frames	Data Loss
PHOENIX14T	7 sec	210×260	192	58	30.21%
How2Sign	41 sec	1280×720	996	86	8.63%

5.2 Experiments

Our objective is to demonstrate that enhancing the quality of input data leads to superior predictions, irrespective of the model employed. The premise is that improving data quality inherently contributes to better overall outputs which the model previously missed (refer to Table 2). Before jumping directly on the integration with EMPATH, or any framework for that matter, our goal is to manually test the acceptance of this particular interpolation model to see how well it works from the ground level. Hence we opted to test the enhancement in real-life data quality through testing, utilizing a pretrained transformer model[1] trained on the Google Isolated Sign Language dataset. This dataset, rich with annotations, provides 250 American Sign Language words. The employed transformer model implements multihead attention [10] and leverages Keras [4] and

[1] kaggle.com/code/markwijkhuizen/gislr-tf-data-processing-transformer-training

Table 2: Highest Prediction Scores (Max 1) for Different Test Cases with Mediapipe and MediaPipe + Algorithm

Ground Truth	**MediaPipe Only Pred**	**Prediction Score (M)**	**MediaPipe +Algo Pred**	**Prediction Score (M+A)**
jump	thankyou	0.04671	jump	0.03349
apple	apple	0.07699	apple	0.09979
duck	duck	0.70121	duck	0.75646
zebra	police	0.07585	police	0.06805
gift	refrigerator	0.02329	gift	0.02303
lamp	helicopter	0.07294	lamp	0.05680
boat	boat	0.04750	boat	0.04712

TFLite [7], ensuring compatibility for deployment on mobile devices. The central evaluation metric focuses on the model's accuracy when presented with new test data generated in-house (see Figure 1).

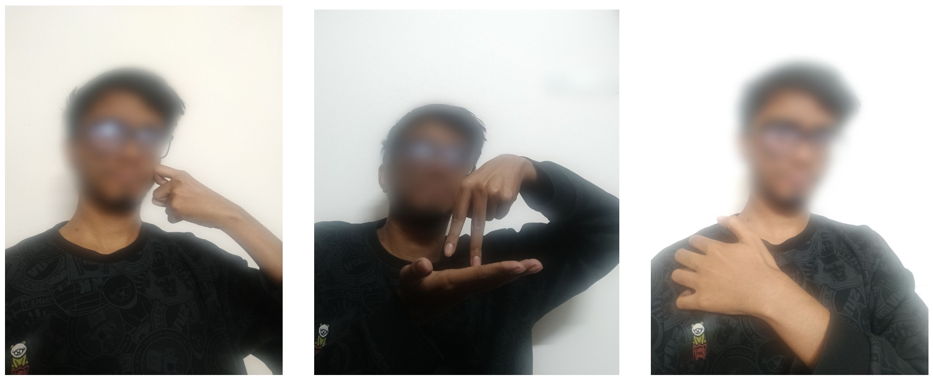


Fig. 1: Sample Test Data (anonymity maintained)

Initially, the model is fed with coordinates detected solely by the MediaPipe Holistic framework. Next, the model is provided with coordinates processed through our algorithm implementation. The key criterion for assessment involves comparing the model's predictions in both scenarios and determining which set of predictions aligns more closely with the ground truth, where MediaPipe coupled with our algorithm yielded **7.692% more prediction accuracy** and **6.297% more confidence**. (see Table 3).

Replicating ASL video samples from the American Sign Language University[2], we introduced challenges like varying lighting, rapid hand movements, and tilted angles, simulating real-world conditions. Noteworthy differences in predictions emerged, particularly in words with swift hand movements or challenging

[2] Lifeprint.com © Dr. William Vicars

Table 3: Accuracy with Average of Highest Scores

Test Cases	**MediaPipe Only Correct Pred %**	**Prediction Score (M)**	**MediaPipe + Algo Correct Pred %**	**Prediction Score (M+A)**
52	76.923	0.16442	84.615	0.17547

data quality for MediaPipe's landmark detection (see Figure 2 & Figure 3). While not consistently accurate (Table 2 zebra case), the algorithm effectively minimizes incorrect predictions, emphasizing correctness. Despite test cases had low prediction scores due to their difficult nature, the algorithm demonstrated a consistent impact on prediction consistency and accuracy across a diverse set of sign language expressions (20%+ of total words). Conducting 500 iterations for each test case provided a robust assessment.

In future works, we plan to integrate the model into live sign language recognition systems to enable more accurate real-time predictions. Further improvements may include expanding the model's ability to handle complex occlusions and exploring ways to generalize the approach across pose and face movements.

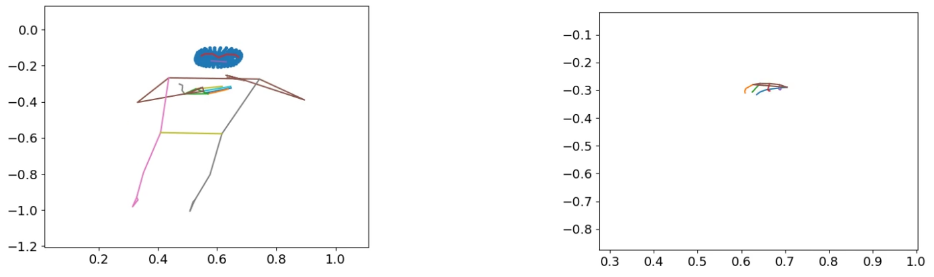
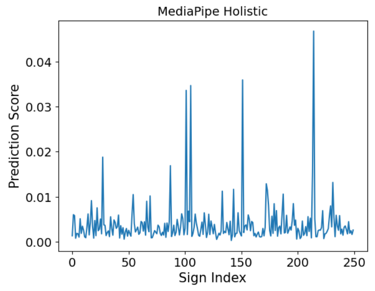
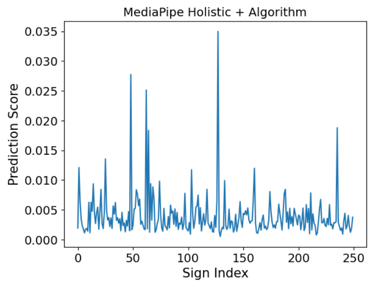


Fig. 2: Missing left hand regenerated to predict movement



(a) thankyou 0.046715081



(b) jump 0.033497258

Fig. 3: Highest prediction scores on a test video with ground truth "jump"

6 Limitations of Development and Deployment

While the EMPATH model has demonstrated excellent results, certain limitations are to be encountered during its deployment in future. Firstly, the availability of dictionary-type training data is currently lacking, especially for languages like Bangla. To extend the model to recognize thousands of words, it is necessary to develop a huge training data, core model trained on large number of parameters and provide access through APIs. This limitation restricts the immediate scalability of the model to handle a broad range of vocabulary in real-world applications.

Additionally, the preprocessing of keypoints, particularly with MediaPipe, has proven to be slow. This is largely due to the unclear documentation regarding GPU utilization. As a result, the preprocessing pipeline currently runs on the CPU, which significantly increases computation time.

References

1. Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., Grundmann, M.: Blazepose: On-device real-time body pose tracking. arXiv preprint arXiv:2006.10204 (2020)
2. Camgoz, N.C., Hadfield, S., Koller, O., Ney, H., Bowden, R.: Neural sign language translation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7784–7793 (2018)
3. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7291–7299 (2017)
4. Chollet, F., et al.: Keras. https://keras.io (2015)
5. Duarte, A., Palaskar, S., Ventura, L., Ghadiyaram, D., DeHaan, K., Metze, F., Torres, J., Giro-i Nieto, X.: How2sign: a large-scale multimodal dataset for continuous american sign language. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2735–2744 (2021)
6. Hasan, K.R., Adnan, M.A.: Empath: Mediapipe-aided ensemble learning with attention-based transformers for accurate recognition of bangla word-level sign language. In: 2024 27th International Conference on Pattern Recognition (ICPR). IAPR (2024)
7. Lee, J., Chirkov, N., Ignasheva, E., Pisarchyk, Y., Shieh, M., Riccardi, F., Sarokin, R., Kulik, A., Grundmann, M.: On-device neural net inference with mobile gpus. arXiv preprint arXiv:1907.01989 (2019)
8. Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.L., Yong, M.G., Lee, J., et al.: Mediapipe: A framework for building perception pipelines. arXiv preprint arXiv:1906.08172 (2019)
9. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
10. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)